DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CALIF 93940

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A METHODOLOGY FOR THE CONVERSION
OF TACTICAL BOARD GAMES TO
COMPUTER ASSISTED WAR GAMES

by

William Clifton Griggs
and
John Arthur Thompson Jr.

September 1980

Thesis Advisor: E. Roland

Approved for public release; distribution unlimited

T196292

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): A Methodology for the Conversion of Tactical Board Games to Computer Assisted War Games	5. TYPE OF REPORT & PERIOD COVERED: Master's Thesis; September 1980
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): William Clifton Griggs and John Arthur Thompson Jr.	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93940	12. REPORT DATE: September 1980
	13. NUMBER OF PAGES: 246
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): Naval Postgraduate School, Monterey, California 93940	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
ASW War game, Anti Submarine Warfare War game, War game, game, board game, tactical war game, tactical board game

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

As part of the development of the capabilities of the Command, Control and Communications (C3) Laboratory at the Naval Postgraduate School, there was a need for an in-house computer assisted tactical war game. The objective of this thesis was to satisfy that need. An anti-submarine warfare board game, UP SCOPE!, which was designed, developed and produced by Simulations Productions, Inc. of New York, was

(Continuation of abstract)

used as the model. This thesis provides an interactive computer assisted anti-submarine warfare war game called Up Scope which is written in FORTRAN. This thesis also develops a framework for any future computer adaptation of a tactical board game, details a players manual and gives full documentation of the computer programming. Program listings, a sample game and several tactical scenarios are also included.

Approved for public release; distribution unlimited

A Methodology for the Conversion of Tactical Board Games to Computer Assisted War Games

by

John Arthur Thompson Jr.
Captain, United States Air Force
B.S., United States Air Force Academy, 1973
M.S., Troy State University, 1978

and

William Clifton Griggs
Lieutenant Commander, Supply Corps
United States Navy
B.S., Miami University, 1971

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September, 1980

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF 93940

ABSTRACT

As part of the development of the capabilities of the Command, Control and Communications (C3) Laboratory at the Naval Postgraduate School, there was a need for an in-house computer assisted tactical war game. The objective of this thesis was to satisfy that need. An anti-submarine warfare board game, UP SCOPE!, which was designed, developed and produced by Simulations Productions, Inc. of New York, was used as the model. This thesis provides an interactive computer assisted anti-submarine warfare war game called Up Scope which is written in FORTRAN. This thesis also develops a framework for any future computer adaptation of a tactical board game, details a players manual and gives full documentation of the computer programming. Program listings, a sample game and several tactical scenarios are also included.

TABLE OF CONTENTS

I. INTRODUCTION ---------- 7

II. DESCRIPTION AND PLAY OF UP SCOPE ---------- 10

A. INTRODUCTION ---------- 10

B. GENERAL COURSE OF PLAY ---------- 11

C. GAME EQUIPMENT ---------- 12

D. DEFINITION OF TERMS ---------- 14

E. GAME INITIATION ---------- 15

F. COMMAND PHASE ---------- 16

G. COMBAT PHASE ---------- 20

H. MOVEMENT PHASE ---------- 26

I. SEARCH AND DETECTION PHASE ---------- 27

J. VICTORY DETERMINATION PHASE ---------- 32

K. STATUS DISPLAY PHASE ---------- 32

III. USER'S MANUAL ---------- 33

A. INTRODUCTION ---------- 33

B. PREGAME REQUIREMENTS ---------- 33

C. LOADING GAME PARAMETERS AND PLATFORMS -------- 37

D. GAME TURNS ---------- 42

E. INPUT OF INSTRUCTIONS ---------- 51

IV. PROGRAM DOCUMENTATION ---------- 54

A. INTRODUCTION ---------- 54

B. DESCRIPTION OF MAJOR ARRAYS ---------- 54

C. DESCRIPTION OF MAJOR VARIABLES ---------- 67

D. DIMENSION, COMMON AND DATA STATEMENTS -------- 71

E. GAME PARAMETERS AND WEAPON SYSTEMS INITIATION -- 72
F. GAME CONTROL ---------- 79
G. DISPLAY SUBROUTINES ---------- 84
H. COMBAT EXECUTION SUBROUTINES ---------- 97
I. UTILITY SUBROUTINES ----------113
V. METHODOLOGY ----------117
A. INTRODUCTION ----------117
B. COMPUTER ASSISTANCE ----------118
C. RULES AND PROCEDURES ----------119
D. DATA STRUCTURE ----------122
E. INPUT AND OUTPUT REQUIREMENTS ----------126
VI. FUTURE ENHANCEMENTS AND CONCLUSIONS ----------128
A. FUTURE ENHANCEMENTS ----------128
B. CONCLUSION ----------130
APPENDIX A. SPECIFIC SCENARIOS ----------132
APPENDIX B. SHIP DATA SUMMARY ----------139
APPENDIX C. WEAPON DATA SUMMARY ----------149
APPENDIX D. AIRCRAFT DATA SUMMARY ----------151
APPENDIX E. PRACTICE GAME ----------153
APPENDIX F. PROGRAM LISTING ----------187
APPENDIX G. PROGRAM ACCESS ----------244
BIBLIOGRAPHY ----------245
INITIAL DISTRIBUTION LIST ----------246

I. INTRODUCTION

All war games are based on some form of a model, and so it is natural that a research effort concerning war gaming be directed toward the development of a model suitable for that purpose. However, this thesis does not develop a model, basically for two reasons. The first is the fact that the authors' interests are in computer applications. The second is the great wealth of models and modelling technology available today. James F. Dunnigan, the president of Simulations Publications Inc. (SPI), writes, "There are two game-design rules that control all others...Keep it simple... (and)...plagiarize." When he says "plagiarize" he means whenever possible use existing technology for there is no reason to recreate what has been done before. The authors took Mr. Dunnigan at his word, and selected an SPI board game entitled "UP SCOPE!" as the war game on which to base their work.

This research effort has two major objectives. The first is to develop an "in house" war gaming capability for the Command, Control and Communications (C3) Laboratory located at the Naval Postgraduate School. This capability is necessary because it is not sufficient for military students to know what types of games exist, the objectives they are expected to accomplish and how to play them. The students

must also be given the opportunity to gain firsthand knowledge about the design, development and programming of war games. The second objective is to examine the problems associated with the conversion of a tactical board game to a "user-friendly" computer-assisted war game. The thesis develops a general outline to follow when a conversion is planned, and discusses the problems encountered during the conversion of "UP SCOPE!."

Chapter II, Description and Play of Up Scope, provides the reader with the background, rules and procedures necessary to play the game. Chapter III is a step by step "player's guide" to the computer version of "Up Scope!." It leads the player through the game by including and discussing key portions of a practice game. The entire practice game is included as an Appendix. Chapter IV is titled 'Program Documentation," and includes detailed documentation of the FORTRAN statements. The program itself uses standard FORTRAN IV, and is presently operating on an IBM 360 computer. The only change required to transport the program is the inclusion of a random number generator subroutine called RANDOM. The IBM 360 located at the Naval Postgraduate School has such a subroutine in its library. It is hoped that Chapter IV will make it easy for the program to become the basis for future research and expansion by other students having an interest in "grass roots" war gaming. Chapter V presents the outline developed for conversion of a board

game to a computer-assisted game, and it discusses the problems encountered and changes made to "UP SCOPE" during the conversion.

It is believed that the outline and discussion of problems encountered included in the thesis and actual play of the computerized version of "UP SCOPE!" will greatly increase the military students awareness of the gaming, and give him a better understanding of what designing, developing and programming a war game entails.

II. DESCRIPTION AND PLAY OF UP SCOPE

A. INTRODUCTION

Up Scope is essentially a computer aided, interactive, tactical simulation of submarine and anti-submarine warfare in the 1980's. It has been adapted from the tactical simulation board game "UP SCOPE!" which was designed, produced and marketed by Simulations Publications Inc., New York, N.Y. "UP SCOPE!" is used with SPI's permission.

This war game furnishes fairly complete, albeit unclassified, information on every significant submarine and anti-submarine platform and weapon system employed by both the NATO powers and the Soviet Union today. Although this game is essentially designed for two players (Red and Blue), any reasonable number can play. As in most war games, the additional players would be relegated to subordinate commander roles. It should be noted that when the game is played with two players, the game is rigid-ruled and self-umpiring. Since this thesis is unclassified, the actual performance parameters for weapon systems, sensors, aircraft, ship and submarine characteristics were derived from unclassified sources and adapted in conjunction with parameters currently implemented in the board game "UP SCOPE!"

B. GENERAL COURSE OF PLAY

Up Scope is basically designed for two players, one of whom normally commands the submarine force while the opposition commands the anti-submarine force. After deciding upon the particular scenario to be played, each player selects his force composition. Appendices B, C and D contain the relevant Ship, Weapon and Aircraft Data Summaries. The following table illustrates the number of surface ship and submarine classes available for selection from each represented country.

Country	Number of Surface Ship Classes	Number of Submarine Classes
Australia	5	1
Canada	5	1
France	16	5
Great Britain	12	5
Italy	11	3
Soviet Union	13	8
United States	28	11
West Germany	4	2

Up Scope is played in a series of successive game turns, each of which consists of a command phase, a combat phase, a movement phase, a search phase, a victory determination phase and a display phase. During the command phase each player privately determines combat, movement (speed and

destination) and search missions for all units under his control.

As this is essentially an anti-submarine warfare game, the only sensors employed are active and passive sonar suits. Radar, ESM and visual detection methods are not pursued.

At the end of the command phase all commands have been entered on the keyboard. The computer then executes combat, movement and search missions for every platform involved. Although the program executes combat, movement and search missions sequentially, they are considered and treated as though they occur simultaneously.

C. GAME EQUIPMENT

The game board is a numbered hexagonal grid pattern superimposed over open ocean. Each "hex" represents a one nautical mile area of ocean. Thus, traversing four hexes is the equivalent of moving four nautical miles. Each hex address is composed of a four-digit number (i.e. XXYY); the XX represents the column coordinate and the YY represents the row coordinate. Figure 1 illustrates the four corners and middle section of the game board.

During the play of Up Scope, reference is made to "adjacent" hexes. Any hex that has a common border with another hex is an adjacent hex. Therefore, each hex except those located on the boundaries has six adjacent hexes. Referring to Figure 1, the hexes containing a $ are adjacent to hex 1622.

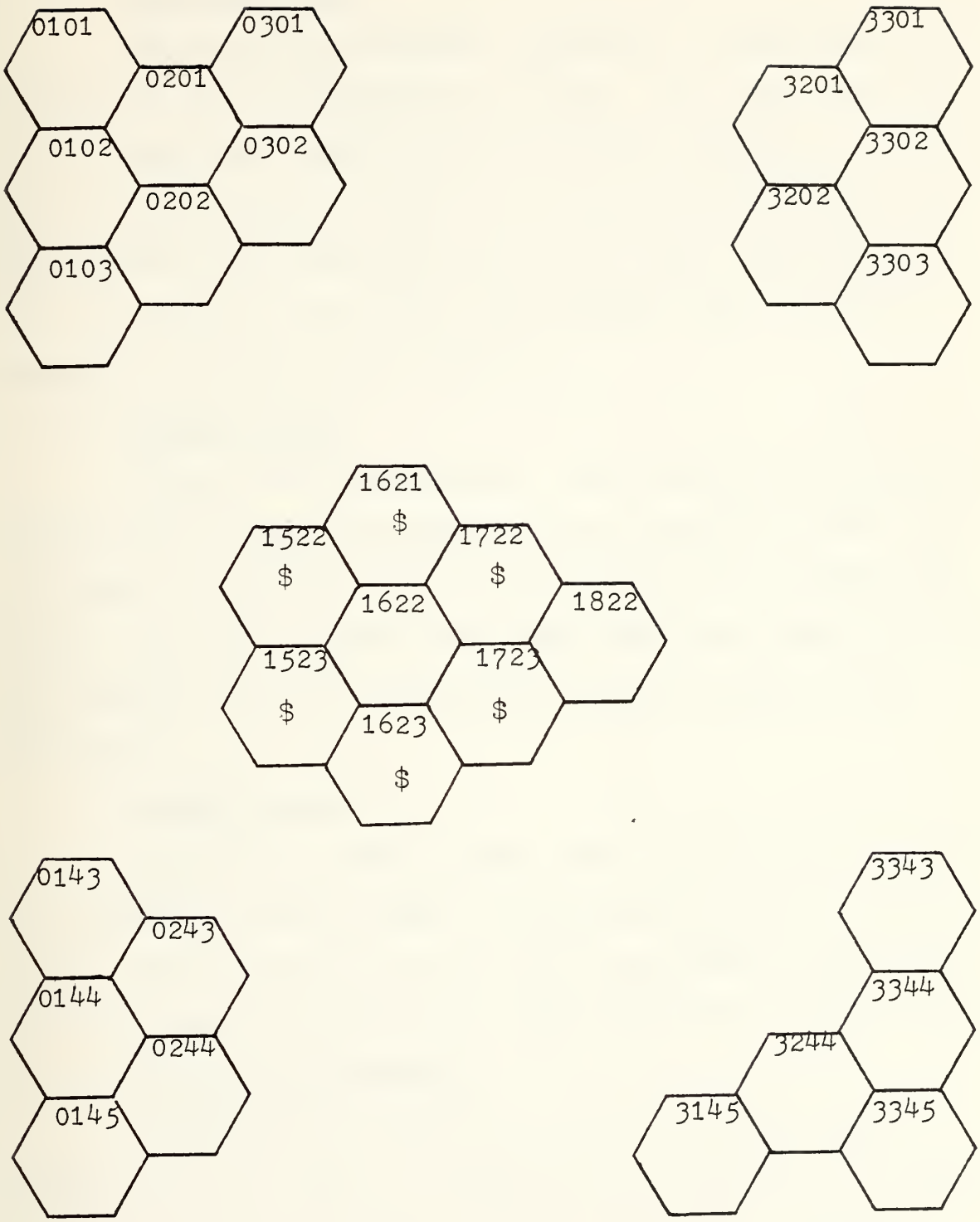

Figure 1. Game Board

D. DEFINITION OF TERMS

1. Platform Class

The Ship Data Summary, Appendix B, categorizes all surface ships and submarines by class. The characteristics of any particular platform are located in Appendix B according to the platform's class. For instance, the Charlie and Papa class Soviet submarines share the same characteristics, and are located in the Ship Data Summary under the Soviet submarine class entitled "Charlie".

2. Speed Command

During the command phase, each platform is assigned a speed command which indicates a rate of speed at which the platform is considered to be moving during the current game turn. The speed command determines the total number of hexes the platform must move during the movement phase of the game turn.

3. Search Command

During the command phase, each eligible platform may be assigned a search command indicating whether the active sonar, passive sonar or both active and passive sonar suits are to be employed during the search phase of the game turn.

4. Search and Detection

Detections by a platform's active sonar, passive sonar or by both will be recorded for an aircraft or surface ship only against submarine targets, and for a submarine against both surface ships and submarines. Detections

presented will include friendly as well as adversary targets. It is up to the player to discriminate which target is which. In general a platform may only attack a platform it detected during the prior game turn.

5. Detection Value

When executing search commands a searching platform is assigned a detection value which indicates the basic ability of its detection equipment to detect and localize previously undetected targets.

6. Evasion Value

When a platform executes a search command all potential targets are assigned an evasion value. This value indicates the platform's basic ability to avoid detection based on both the platform's inherent characteristics and/or range, ocean and water conditions which aid the platform's ability to evade detection.

E. GAME INITIATION

Appendix A lists several scenarios, each of which constitutes a distinct game. Players may select any of those scenarios or develop their own scenarios as they desire. Initializing the game requires each player to select a password which enables the player to enter commands and retrieve relevant information regarding combat operations, damage reports and search results of platforms under his command. The player then selects his force composition (i.e. three surface ships and two submarines), gives each

platform a name, identifies the platform class (Spruance, Nimitz, Krivak, Charlie, etc.) and designates the initial position on the game board. After entering this initial information the game is ready to begin. The game consists of game turns, each of which represents six minutes of elapsed time. Game turns are executed independently, the only exception is the fact that a scheduled combat mission can succeed only if that particular platform detected the target on the previous game turn. Game turns are then executed until the game ends, either by preset victory conditions being met, the preselected number of game turns having been executed or mutual decision by both Red and Blue players. Figure 2 displays a block diagram of the phases of Up Scope.

The availability of aircraft is restricted to ship-based aircraft carried aboard the combatants involved in the scenario. Both fixed wing aircraft and helicopters are available. The Ship Data Summary, Appendix B, lists type and number of aircraft carried aboard the various classes of surface ships. The Aircraft Data Summary, Appendix D, lists the detection values, maximum speed, weapon system mix and resupply ratings for each type aircraft. Some aircraft may carry either of two different weapon systems. The player must decide before launching the aircraft which weapon system he desires to employ. An aircraft weapon load cannot be changed after launch.

F. COMMAND PHASE

After reviewing the previous game turn's displays of combat action results, damage suffered by friendly forces,

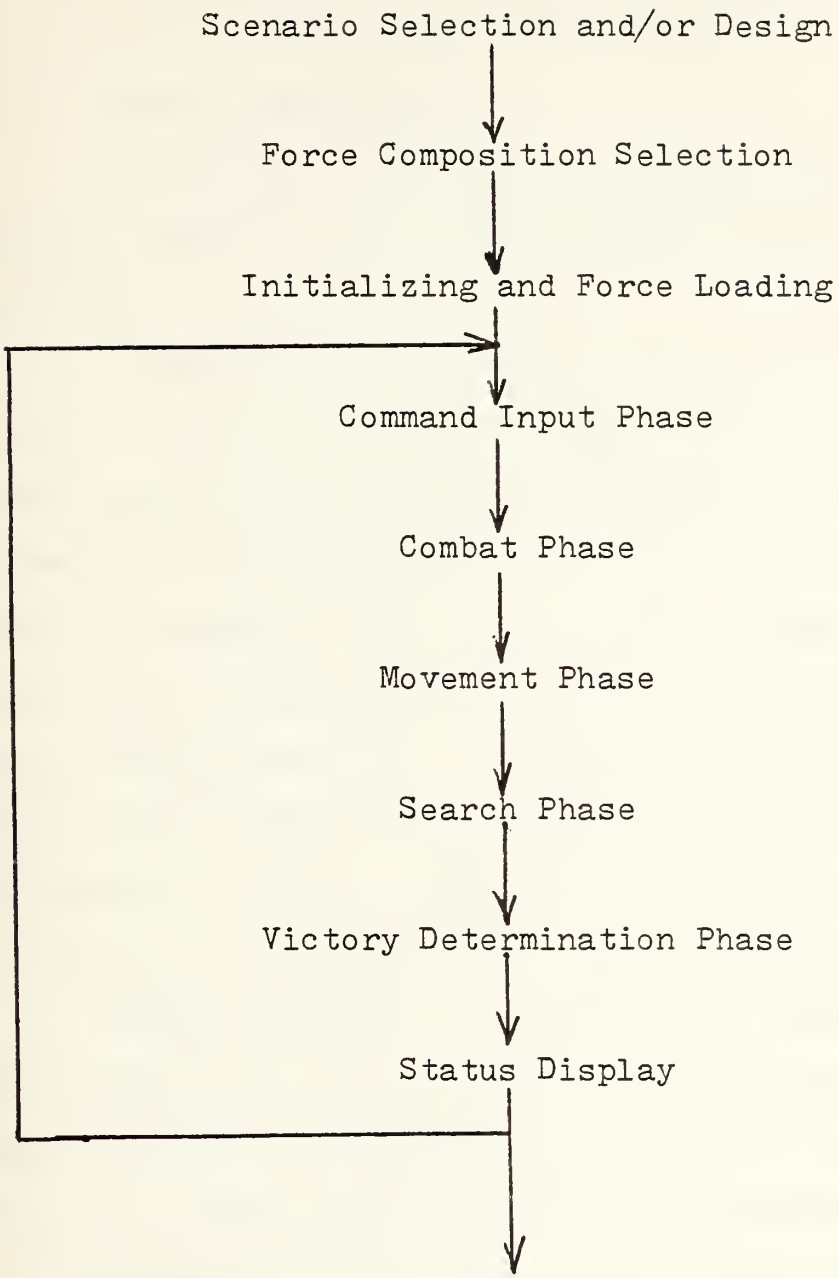

Figure 2. Up Scope Block Diagram

current platform locations and detections resulting from the previous game turn searches, the player privately determines his current game turn strategy.

1. Combat Operations

Combat operations are subdivided into two categories: Aircraft combat operations and surface ship/submarine combat operations.

Aircraft can only release one weapon system per game turn and only at a submarine target. The player selects either weapon system number one or weapon system number two. The weapon systems carried are defined in the Aircraft Data Summary, Appendix D. The weapon system listed first is weapon system number one and the system listed second is weapon system number two. Therefore, to engage a target, the player enters weapon system number one or two and the target's hex address location. Aircraft cannot be attacked during any scenario under the present rules of Up Scope.

Surface ships and submarines may engage a single target with one or more of their weapon systems simultaneously; therefore, the player may enter more than one combat mission for each platform in the manner described above. Due to the nature of the game, there is no interference between weapons in a multiple weapon system attack. The weapon systems carried by each platform are listed in the Ship Data Summary, Appendix B. Weapon system numbering is in the order they are listed in Appendix B.

2. Movement

The player must keep in mind that the number of hexes a platform traverses during each game must be consistent with the current maximum speed allowed and that the speed must equal the distance to the destination hex. The player enters projected speed and destination hex address each game turn.

At the start of a scenario, all aircraft are considered aboard their respective surface ships and may not move independently until they are launched. During each command phase, a player may launch any single aircraft from each capable surface ship under his control. After launch the aircraft move according to the established movement rules. Aircraft may not be launched once it has been determined the parent ship is sunk. Also, if a parent ship departs the board boundary, the aircraft must immediately depart the board boundary.

3. Search

The player may not assign a search mission to any aircraft that is either not flying (still based on the parent ship) or is transiting to a new destination hex. To execute a search mission the aircraft must have a speed of zero for that game turn, which signifies that a fixed wing aircraft is loitering in the area or that a helicopter is either hovering or loitering in the area. Additionally, aircraft may not be assigned both active and passive search missions simultaneously.

For surface ships and submarines, only platforms with a one in the column labeled "SS" in the Ship Data Summary, Appendix B, may execute both active and passive search missions simultaneously. The player, when asked in the game turn input phase, enters a zero for no search mission, a one for an active search mission, a two for a passive search mission and a three for both active and passive search missions simultaneously. If a platform is not assigned a detection value under active and/or passive detection in the Ship Data Summary, it may not be assigned a search mission employing that detection capability. For example, the Nimitz class aircraft carrier possesses neither type of detection value and thus cannot execute a search command. Aircraft may not be assigned a search mission if the aircraft is assigned a move command other than "none."

The assignment of search commands is at the player's discretion. A platform is never required to execute a search mission.

G. COMBAT PHASE

After completion of the command phase and players have input their combat, movement and search commands, the program shifts into execution. The first phase is the combat phase which executes all combat orders.

1. Introduction

Contemporary weapon systems include a large variety of ship-to-ship missiles, sophisticated wire-guided and/or

homing torpedoes, missile launched depth bombs/torpedoes and depth charges. The Ship Data Summary, Appendix B, along with the Weapon Data Summary, Appendix C, indicate all the following information.

a. Weapon type

The various weapon types with which each platform class is equipped. Surface ships and submarines carry a maximum of three different weapon systems.

b. Accuracy Rating

Each weapon type has an accuracy rating ranging from six (best) to three (worst).

c. Maximum Range

Each weapon system has a maximum effective range expressed in terms of number of hexes it will traverse.

d. Weapon Resupply

The number of weapon delivery units available for each weapon system, ranging from one to six. A weapon delivery unit is composed of the number of individual weapons normally expended in a single attack utilizing that weapon system. For example, if a Charlie class submarine would normally expend all eight SSN-7 missiles in an attack, then the number of weapon delivery units would be one, and it would be composed of eight SSN-7 missiles.

e. Defense Rating

The Defense rating of each surface ship/submarine class, range from three (best) to zero (worst). It is a

measure of how capable the platform is in defending against attacks and/or evading the attacks.

f. Aircraft

The number and type of aircraft each surface ship class carries and is capable of employing in an anti-submarine role.

2. Execution of Combat Missions

Combat missions are executed during the game turn according to the following restrictions.

a. Resolution

Each combat mission is resolved separately against the target platform using the procedure given below.

b. Attacking Platform

Each weapon system type may be employed only once during a game turn. Different weapon systems aboard an attacking surface ship/submarine may only be employed against the same target. (i.e., a platform may not attack two or more targets during the same game turn).

c. Target

An attacking platform must have detected the target on the previous game turn. A weapon may be employed against a undetected target; however, a hit cannot be obtained.

d. Damage

All combat is considered to be simultaneous. All damage inflicted occurs at the end of the game turn and does not affect current game turn commands.

e. Resupply

Once all reloads of a weapon system have been expended, the weapon system cannot be employed. There is no external resupply of any weapon system during a game.

3. Resolution of Combat Actions

Damage inflicted due to combat resolution takes effect at the end of the game turn; thus, a platform that has been sunk or hit is able to execute and resolve all assigned combat missions. For each combat mission the program checks to insure the target is within weapon range, cross references the accuracy rating of the employed weapon with the defense rating of the target platform, draws a random number to simulate rolling a die and locates the resulting random number (an integer from one to six) under the proper resolution column on the Combat Results Table, Figure 3. If the indicated result is not an "H" (hit), then the attack failed. If the result is an "H", then the attack was successful and the durability value of the target is reduced by one.

4. Weapon System Resupply

Each weapon system carried by a platform has a weapon system resupply rating which indicates the comparative supply and reload facilities available for that weapon system on that platform. Each time a given platform executes a combat mission, the corresponding resupply rating for that weapon system is decremented by one. Once a resupply rating is

		Defense Rating of Target Platform					
Weapon Accuracy Rating	6 5 4 3	0	1 0	2 1 0	3 2 1	3 2	3 any
Random Number Drawn	1	H	-	-	-	-	-
	2	H	H	-	-	-	-
	3	H	H	H	-	-	-
	4	H	H	H	H	-	-
	5	H	H	H	H	H	-
	6	H	H	H	H	H	H

Figure 3. Combat Results Table

reduced to zero that platform may not be assigned a combat mission employing that weapon system.

5. Platform Damage

The durability value assigned to each surface ship/submarine indicates the number of hits required to sink it. A submarine which has a durability value of two would require two separate hits to sink it. The initial durability value for each type of platform is indicated below.

Type of Surface Ship/Submarine	Initial Durability Value
Aircraft carrier	3
Cruiser, Submarine	2
Destroyer, Frigate	1

a. Damage Assessment

For each durability point of damage suffered, the platform's maximum speed, weapon system resupply ratings, active detection value and passive detection values are reduced by one.

b. Repair

Damage may never be repaired during the scenario.

c. Sunk

When a platform's durability point value is reduced to zero, the platform is considered sunk.

6. Disturbed Water

Whenever a combat mission results in a weapon detonation, disturbed water is assumed to occupy the hex in which the target is residing and the six adjacent hexes as well. Detonations will not occur when a weapon system misses the target and the weapon is a torpedo; however, depth charges and depth bombs will detonate regardless of a hit or miss. The effects of disturbed water are discussed later in this chapter.

H. MOVEMENT PHASE

After completion of the combat phase during which all the combat orders are executed, the program shifts to the movement phase and all movement commands are executed.

The speed command given a platform during the command phase determines the total number of hexes it must traverse during the current game turn. It also must match the hex distance between the current location and future location. Any platform may be assigned a speed of zero for any game turn. Aircraft must have a speed of zero in order to conduct a search mission. Also surface ships and submarines, in general, have better detection capabilities at lower speeds than at higher speeds.

The maximum speed is limited to the maximum for that class of platform as indicated in the Ship Data Summary, Appendix B. Also, the maximum speed will be permanently

reduced as a result of damage sustained in combat operations.

I. SEARCH AND DETECTION PHASE

After completion of the movement phase, the program shifts to the search and detection phase, during which the search commands previously entered are executed for all platforms.

1. Introduction

All platforms (surface ships, submarines or aircraft) in this game employ only detection capabilities which simulate the use of both active and passive advanced sonar equipment. The detection and evasion capabilities for all platforms are delineated in the Ship and Aircraft Data Summaries in appendices B and D respectively. This includes the following:

a. Active Detection Value

The platform's active detection value (ADV) simulates the capability of its active sonar system(s). The ADV ranges from five (best) to one (worst).

b. Passive Detection Value

The platform's passive detection value (PDV) simulates the capability of its passive sonar system(s). The PDV ranges from five (best) to three (worst).

c. Evasion Value

Surface ship's and submarine's evasion values (EV) which simulate the platforms radiated noise level. The

higher the EV value, the quieter the platform is considered to be and therefore the more difficult it is to locate using a passive search during the game turn. The EV rangesfrom five (best) to one (worst).

d. Simultaneous Active and Passive Search Missions

If a platform has a one in the "SS" column for its class in the Ship Data Summary, Appendix B, then the platform is capable of executing both active and passive search missions simultaneously during the game turn.

e. Speed Degradation

Normally, sonar system performance is degraded the faster the surface ship or submarine is traveling, due to increased water flow noise and the own ship's radiated noise level. Some more advanced sonar systems have solved this degradation to some degree. In Up Scope if a platform has a one in the "SD" column for its class in the Ship Data Summary, Appendix B, this indicates that when a platform executes a passive search mission there is neither degradation in capability nor reduction due to platform speed in the passive detection value.

2. Search Mission Resolution

Search missions assigned during the command phase are resolved during this search and detection phase. The program computes the appropriate active or passive detection value of the searching platform, taking into account all the appropriate modifiers, then calculates evasion values for all other

platforms except aircraft. The program then compares the evasion value of each platform to the final adjusted detection value of the searching platform. If the detection value is greater than or equal to the evasion value, the target platform is considered detected by the searcher. If the evasion value is greater than the detection value then no detection has been made.

a. Active/Passive Detection Value Computation

When a platform executes a search mission, its appropriate detection value (ADV or PDV) is cumulatively modified by the following criteria:

(1) The platform's ADV/PDV is reduced by one if its speed is one or two.

(2) The platform's ADV/PDV is reduced by two if its speed is three or four.

(3) The platform's ADV/PDV is reduced by two if the water condition has been set as "erratic," discussed later in this chapter.

(4) The platform's ADV/PDV is reduced by three if the platform occupies a disturbed water zone, discussed later in this chapter.

(5) The platform's ADV/PDV is reduced by one for each durability point of damage sustained by the platform.

(6) As already mentioned, the platform's ADV/PDV is not reduced if the platform's speed is zero, or if column "SS" for the class in the Ship Data Summary, Appendix B, has a one on it. Otherwise, the ADV/PDV is reduced as stated above.

(7) A platform's ADV/PDV cannot be reduced below zero.

b. Active Evasion Value (AEV) Computation

For every active search mission being executed, the program computes an AEV for every potential target in the following fashion (This computation does not make use of the evasion value listed in the Ship Data Summary, Appendix B.):

(1) The basic AEV is determined by computing the hex distance between the two platforms, dividing the result by three and rounding up. The following Table illustrates the computation.

AEV	Hex Distance
1	0 - 3
2	4 - 6
3	7 - 9

(2) The basic AEV is increased by three if the platform occupies a disturbed water zone.

c. Passive Evasion Value (PEV) Computation

For every passive search mission being conducted, the program computes a PEV for every potential target in the following fashion:

(1) The platform's basic PEV is determined by computing the hex distance between the two platforms, dividing the result by ten and then rounding up. The following illustrates the computation:

PEV	hex distance
1	0 - 10
2	11 - 20
3	21 - 30

(2) The platform's PEV is increased by the EV indicated for its class in the Ship Data Summary, Appendix B.

(3) The platform's PEV is increased by three if it occupies a disturbed water zone.

(4) The platform's PEV is decreased by two if the platform is assigned an active search mission for the current game turn.

(5) The platform's PEV is decreased by three if the platform executed a combat mission during the game turn.

(6) The platform's PEV is automatically set at six if the platform remained stationary, was not assigned an active search mission and did not execute a combat mission during the current game turn.

3. Water Conditions

During any scenario, the water conditions are prescribed by the scenario rules as either "Normal" or "Erratic" and this information is set during the initializing and force loading phase. It should be recognized that if the water conditions are erratic during a scenario, the effectiveness of both active and passive sonar suits is degraded.

Disturbed water zones are created by the detonation of weapons. The zone consists of the hex in which the weapon

detonated and the six adjacent hexes. The disturbed water zones exist only during the current game turn. If occupied by a searching platform the zone decreases detection values and if occupied by potential targets, the evasion value is increased as just described.

J. VICTORY DETERMINATION PHASE

After completing the search phase, the program enters the victory determination phase. If all of either player's ship forces have been sunk, the program exists after displaying the final results. If either the predetermined number of ships designated during the initializing and force loading phase have been sunk, or the designated number of game turns have been completed, the program attempts to end the game. It queries the players to see if they desire to continue or, in fact, desire to exit the game.

K. STATUS DISPLAY PHASE

During the status display phase the player reenters the game by requesting the various status displays available regarding damage inflicted, damage suffered, search results and individual platform status. Then the player enters the command input phase.

III. USER'S MANUAL

A. INTRODUCTION

The computerized version of Up Scope is an easy game to play even without a complete understanding of the rules discussed in Chapter II. All the difficult chores, such as computation of detections and damage, and all bookkeeping tasks are performed by the computer. Through the interactive nature of the game, a player is led through the various stages, giving all the information the player needs to know to make decisions. However, a general understanding of the rules and game procedures is required to use the board and Appendices. An understanding of the game also allows the players to play more effectively.

This chapter includes a section on pregame requirements which deals with the development of scenarios. It also includes several sections containing the instructions needed to use the computer program. The instructions are based on excerpts from the practice game in Appendix E. Instructions for obtaining a copy of the program deck are contained in Appendix G.

B. PREGAME REQUIREMENTS

The pregame requirements for Up Scope can be as simple or as complicated as the players desire. However, as a

minimum, the following steps in developing a scenario should be taken to ensure a successful game.

1. Objective

Every game played should have an objective. This could mean a detailed write-up, with a general scenario to set the world's political and economic situation and a specific scenario setting the tactical situation. It could also be a simple agreement between the players to "search and destroy." Whatever the purpose, both players should agree to the basic objective of the game before they start.

The objective should also include victory determination criteria. This is extremely important because it forces the players to concentrate on setting and achieving the objective. Again the criteria could be simple, such as "fight until one force is totally destroyed", or complex. A complex criteria might be a force level percentage after a specified number of game turns. Appendix A presents several scenarios modified from examples included in the SPI version of "UP SCOPE!".

2. Game Environment

Another important aspect of the scenario is the environment in which the game is to be played. In the computerized version of Up Scope, this includes the size of the ocean area and the water conditions.

a. Game Board

In the board game, SPI supplies the players with two 33 by 45 hex boards. The boards can be placed side by side or lengthwise to increase the size of the playing area. The available size in the computer game is physically unrestricted as long as the number of hexes in the X or Y direction does not exceed 99. When determining the size, the objective of the game and the speed of the ships must be considered. For example, it would take the fastest ship 11 turns just to cross the 33 hex width of the SPI board. Unless the players are willing to spend a great deal of time maneuvering or the scenario dictates otherwise, the size of the area should be held to a minimum.

Up Scope does not restrict the players from moving their forces outside the agreed upon playing area. A "gentlemen's agreement" between the two players must exist to forestall "out of bounds movement."

b. Water Condition

The second environmental factor in Up Scope is the water condition which was discussed in chapter II.I.3. The water condition can be "normal" or "erratic."

3. Game Length

The game can be terminated by any of three methods. Two of them, game turns and number of ships destroyed, are scenario dependent. The third is total destruction of a player's ship forces.

a. Game Turns

The number of game turns to be played is mutually agreed on by the players. It can be used in two ways. The first is actually determined by the scenario, which might involve the Blue forces defending a certain area for a given number of turns. The second use is to put an upper limit on the length of the game. In either case it is preferable to use a relatively small number because the players are given the option of continuing the game beyond the preselected number of game turns.

b. Disengagement

Both players are given the opportunity to prematurely end the game by disengaging when a certain number of their ships have been destroyed. Again, a relatively small number should be chosen because the player will be given the option of continuing when his disengagement criterion has been reached.

c. Total Destruction

Total destruction of a player's forces is the only means of terminating the game without the player's consent. When this occurs, the program will print the final results, and give the players an opportunity to start a new game.

4. Forces

Perhaps the major pregame task, other than determination of the objective, is to decide the forces to

be employed by the Blue and Red players. The number and structure of the forces should be consistent with the objective. Normally the forces should be balanced between the players so that the outcome is determined by tactics and not by superior numbers.

Ship Data Summary, Appendix B, contains all the ship classes in the Up Scope arsenal and all the required information about each class. The Appendix must be consulted when determining the number of ships because it contains the number of aircraft carried by each class. This is important because the current version of the game can only support 100 platforms, including aircraft, and the game loads aircraft automatically for each ship class. Prior to accessing the game, the players should ensure that their combined number of aircraft, surface ships and submarines does not exceed 100.

C. LOADING GAME PARAMETERS AND PLATFORMS

The loading portion of Up Scope consists of several sections; each designed to ease the entry of game inputs.

1. Menu Selection

The first display which the players see when Up Scope is accessed is:

```
EXECUTION BEGINS...
THE FOLLOWING OPTIONS ARE AVAILABLE TO YOU:
1.  ZERO ANY PREVIOUSLY ESTABLISHED DATA BASE.
2.  EXIT THE GAME.
```

3. INITIALIZE GAME PARAMETERS AND COMMENCE PLAY.

ENTER AN OPTION.

>3

Option 1 reinitializes any user inputs from previous games, which allows the players to start a new game without exiting and restarting the program. Options 2 and 3 are self-explanatory. If Option 1 is selected, the players will automatically be returned to menu selection at its completion.

2. Entry of Game Parameters

a. Basic Parameters

Once Option 3 has been selected the program will begin to ask for the basic game parameters. First, the following warning concerning the total number of platforms is printed with a request for the number of game turns.

```
YOU ARE NOW GOING TO ENTER THE BASIC GAME PARAMETERS.
ONE WORD OF CAUTION.  THE PRESENT GAME CAN ONLY HANDLE A
TOTAL OF 100 WEAPON SYSTEMS.  INCLUDED IN  THESE SYSTEMS
ARE SURFACE SHIPS, SUBMARINES AND AIRCRAFT.  THE NUMBER OF
AIRCRAFT ARE DETERMINED BY THE SHIP CLASS AS REFLECTED IN THE
DATA BASE, AND WILL BE LOADED AUTOMATICALLY WHEN YOU SELECT
THE SHIP CLASS.  PRIOR TO BEGINNING THE LOADING PROCEDURE
YOU SHOULD ENSURE THAT THE TOTAL NUMBER OF EACH PLAYERS
WEAPON SYSTEMS DOES NOT EXCEED   50.

ENTER THE NUMBER OF GAME TURNS YOU WANT TO PLAY.
>2
```

After game turns have been entered, the program will print the following request for the water condition.

```
ENTER THE WATER CONDITION (0-NORMAL, 1-ERRATIC).
>0
```

b. Blue/Red Game Parameters

The following lines begin the entry of the Blue player's game parameters by requesting a Blue password. This player has selected "BLUPAS".

```
THE FOLLOWING INPUTS APPLY ONLY TO THE BLUE PLAYER
THE FIRST IS THE BLUE PLAYERS PASSWORD WHICH IS
A MAXIMUM OF 6 CHARACTERS.  THE PASSWORD IS IMPORTANT BECAUSE
IT IS REQUIRED PRIOR TO THE INPUT OF COMBAT, MOVE AND SEARCH
COMMANDS.  IF YOU FORGET THE PASSWORD, YOU WILL HAVE TO RETURN
TO MENU SELECTION, AND RESTART THE GAME.


SELECT AND ENTER YOUR PASSWORD
>blupas


YOUR PASSWORD WILL BE -BLUPAS- .  IF YOU ARE
READY TO CONTINUE TYPE Y(ES).  TYPE N(O) IF YOU WISH TO
CHANGE YOUR PASSWORD.
>y
```

Next the program will ask for the number of Blue surface ships and submarines, and the number of ships that must be destroyed for disengagement to occur. The format as seen below echoes the inputs and gives the player the opportunity to change his choices.

```
ENTER THE NUMBER OF BLUE SURFACE SHIPS.
>2


ENTER THE NUMBER OF BLUE SUBMARINES.
>1


ENTER THE NUMBER OF BLUE SHIPS THAT MUST
BE DESTROYED FOR PREMATURE GAME ENDING.
>2


ECHO
```

```
BLUE SURFACE SHIPS - 2
BLUE SUBMARINES - 1
NUMBER OF BLUE SHIPS DESTROYED TO END GAME - 2

TYPE Y(ES) IF THESE VALUES ARE CORRECT, OR N(O) IF
YOU WANT TO REENTER THEM.
> y
```

If the Blue player does not elect to change his inputs, the program will advance to the Red player's game parameters which is essentially a repeat of the above. In this practice game, the Red player has entered two submarines of which the destruction of one will cause disengagement (not shown).

c. Loading Platforms

Up Scope leads each player through the loading procedure one ship at a time. It will first ask for the player's password. If the password is entered correctly, the player will input a name (six characters or less) for the ship, the ship's class and its initial location. The program automatically loads any aircraft carried by a class of ship, and asks the player for aircraft names. Inputs for the practice game are shown below.

```
BLUE SURFACE SHIP NUMBER     1

ENTER A 6 CHARACTER NAME FOR THE SHIP.
>tincan

ENTER THE SHIP CLASS OF TINCAN
>spruance

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF TINCAN
>500
```

In the above example the hex address has been entered as "500" instead of "0500". The program treats the hex address as a four digit integer and pads leading blanks with zero's. In the following example the entry is "0520".

```
THE SPRUANCE    CLASS OF SHIP HAS 1 SEA SPRITE   TYPE AIRCRAFT.

ENTER A 6 CHARACTER NAME FOR AIRCRAFT  1
>sprul

BLUE SURFACE SHIP NUMBER  2

ENTER A 6 CHARACTER NAME FOR THE SHIP.
>escort

ENTER THE SHIP CLASS OF ESCORT
>knox

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF ESCORT
>0520


THE KNOX        CLASS OF SHIP HAS 1 SEA SPRITE  TYPE AIRCRAFT.

ENTER A 6 CHARACTER NAME FOR AIRCRAFT  1
>knox1

BLUE SUBMARINE NUMBER  1

ENTER A 6 DIGIT CHARACTER NAME FOR THE SUBMARINE.
>lapon

ENTER THE SUBMARINE CLASS OF LAPON
>sturgeon

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF LAPON
>510

YOU ARE NOW READY TO LOAD THE RED WEAPON SYSTEMS.

ENTER THE RED PASSWORD.
>redpas

RED SUBMARINE NUMBER  1
```

```
ENTER A 6 DIGIT CHARACTER NAME FOR THE SUBMARINE.
>ivan

ENTER THE SUBMARINE CLASS OF IVAN
>charlie

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF IVAN
>1510

RED SUBMARINE NUMBER   2

ENTER A 6 DIGIT CHARACTER NAME FOR THE SUBMARINE.
>boris

ENTER THE SUBMARINE CLASS OF BORIS
>november

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF BORIS
>1520
```

D. GAME TURNS

Once the players have entered the game parameters, they are ready to enter the gaming phase.

1. Displays

The first step of each game turn is to review the results of the previous turn and enter combat, move and search instructions. These tasks are controlled by a group of routines labeled STATUS. After the last Red submarine has been loaded, the program will ask for the Blue player's password and print the following menu of STATUS display options:

```
BLUE CALL TO STATUS ROUTINES.  ENTER THE BLUE PASSWORD.
>bluepas
```

THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:

1. DETECTIONS OF ENEMY PLATFORMS.
2. RESULTS OF WEAPONS FIRED.
3. DAMAGE TO FRIENDLY PLATFORMS.
4. STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.
5. BEGIN NEXT GAME TURN.

SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4

If the current game turn is the first turn, the first three options will contain no information, and should be ignored.

a. Detection of Enemy Ships

This option will print the detections of each platform of the calling side in the following format:

DETECTING PLATFORM	SONOR TYPE USED	SHIP TYPE DETECTED	LOCATION
TINCAN	ACTIVE	SUBMARINE	512
ESCORT	ACTIVE	SUBMARINE	512
ESCORT	ACTIVE	SUBMARINE	1522
TINCAN	PASSIVE	SUBMARINE	1522
LAPON	PASSIVE	SURFACE	502
LAPON	PASSIVE	SURFACE	720
LAPON	PASSIVE	SUBMARINE	1511
LAPON	PASSIVE	SUBMARINE	1522

END OF DETECTION ROUTINE. TYPE Y(ES) TO CONTINUE.
>y

b. Results of Weapons Fired

Option 2 examines each friendly platform, and gives the probability of a hit for each weapon fired during

the previous turn other than torpedoes. The result for torpedoes is either "DETONATION" or "NO DETONATION". The following is one example from the practice game:

```
SHIP - ESCORT    COMBAT LOCATION - 1522
WEAPON 2   TYPE - MK 46     PROB(HIT)/DETONATION - DETONATION


SHIP - LAPON     COMBAT LOCATION - 1522
WEAPON 1   TYPE - SUBROC    PROB(HIT)/DETONATION - 0.6667
WEAPON 2   TYPE - MK 48     PROB(HIT)/DETONATION - DETONATION

END OF WEAPON RESULT DISPLAY.  TYPE Y(ES) TO CONTINUE.
>y
```

Some discussion is offered for some of the displays listed above in this section. Above, the Lapon, a sturgeon class nuclear attack submarine, released a MK 48 torpedo and a Subroc against a target located at 1522. The 'WEAPON 1 TYPE-SUBROC' statement illustrates the fact that some weapons detonate regardless of proximity of target, while others (notably torpedoes), only detonate when in close proximity, or when actually hitting a target. In either case, it is classified as a hit. Therefore, when engaging with a "non-torpedo" type of weapon, the player is not sure if the target was hit or destroyed. All that is displayed is the probability of a hit. The player should be aware that torpedo type weapons, on the other hand, either hit (detonate) or miss (no detonation).

c. Damage to Friendly Platforms

This option will give the player the number of hits each ship has suffered if it is not sunk. Otherwise "SUNK" is printed as below:

```
 SHIP        STATUS/HITS
TINCAN            0
ESCORT          SUNK
LAPON             0
END OF DAMAGE DISPLAY ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y
```

d. Status of Friendly Platforms and Input of Instructions

This option displays individual platform status and accepts and processes all relevant commands. The program is descriptive and self-prompting. There are differences between aircraft and other platforms; therefore, there are two slightly different routines. Aircraft must be launched from the parent ship, must have a speed of zero to enable search missions and they must only carry one weapon system during a flight.

As this option is entered, the following is displayed:

```
YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT, MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.
```

(1) Initial display for non-aircraft platforms

The player will see the following and make the entries in the same fashion as in this example:

```
>tincan
```

```
TINCAN IS LOCATED AT POSITION  500 SPEED  0
IT WAS NOT ASSIGNED A SEARCH MISSION
TINCAN WEAPON SYSTEM(S)
     TYPE      RANGE
1    MK 46      20
2    ASROC       7

NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION

>n

TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION     IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>502


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>3

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
         NONE SCHEDULED
DESTINATION 502    SPEED 2
ACTIVE AND PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y

ENTER THE NAME OF THE DESIRED PLATFORM
```

The lines starting with the '>' are the player's entries. The player should recognize that when an active search mission and/or a combat mission is assigned, the platform's risk of being detected is significantly greater.

(2) Full display for non-aircraft platforms. If a platform had combat and search missions scheduled during the previous game turn, the results of these missions are reported just before new input instructions are asked for. The following illustrates the full range of information displays; i.e., Combat results; damage inflicted; movement and search missions undertaken and subsequent orders for the present game turn:

```
>lapon
LAPON IS LOCATED AT POSITION 712 SPEED 2

     THE LAPON ATTACK ON TARGET LOCATED AT 1522

RELEASE OF SUBROC     RESULTED IN
A DETONATION WITH PROBABILITY
OF A HIT OF 4 OUT OF 6

RELEASE OF MK 48      RESULTED IN
A DETONATION(HIT)

IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
    TYPE      LOCATION      METHO      RANGE
SUBMARINE       1311        ACTIVE       6
LAPON WEAPON SYSTEM(S)
    TYPE      RANGE
1  SUBROC       25
2  MK 48        25

NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION

>1311
>1
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)

>2

TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)

>n
```

```
TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION          IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>714

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>1

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        SUBROC       WEAPON      TGT LOCATION  1311
        MK 48        WEAPON      TGT LOCATION  1311
DESTINATION  714     SPEED 2
ACTIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>n
```

(3) Initial Display for Aircraft. The following illustrates the initial display for an aircraft prior to launch:

```
>knox1

KNOX1  IS CURRENTLY NOT FLYING - IT IS ASSUMED YOU
DESIRE TO LAUNCH.


KNOX1  IS NOW BEING LAUNCHED


MOVEMENT NOT DESIRED ENTER N(ONE)  OTHERWISE ENTER
SPEED
DESTINATION HEX       IN THIS FASHION

>12
>1720
```

```
ENTER SEARCH MISSIONS AS FOLLOWS:
    0  NONE SCHEDULED
    1  ACTIVE MISSION
    2  PASSIVE MISSION


>0
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
NONE SCHEDULED
DESTINATION 1720 SPEED 12
THERE IS NO SEARCH MISSION SCHEDULED


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y

ENTER THE NAME OF THE DESIRED PLATFORM
```

(4) Full Display for Aircraft. The following illustrates the difference between the initial display and one with combat results.

```
>knox1

KNOX IS LOCATED AT POSITION 520 AT SPEED 0
IT RELEASED A MK 46  AT A SUBMARINE TARGET LOCATED
AT POSITION  518
THE WEAPON
         DETONATED (HIT)
IT CONDUCTED AN ACTIVE SEARCH WITH FOLLOWING RESULTS
SUBMARINE CONTACT LOCATION      RANGE
           518                    2
KNOX1 WEAPON SYSTEM(S)
     TYPE     RANGE
1 MK 46         20


NO COMBAT ENTER N(ONE)   OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n
MOVEMENT NOT DESIRED ENTER N(ONE)  OTHERWISE ENTER
SPEED
DESTINATION HEX          IN THIS FASHION
```

```
>n

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION


>1
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
NONE SCHEDULED
SPEED IS ZERO AT LOCATION   520
AN ACTIVE SEARCH MISSION WILL BE CONDUCTED


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y
```

The player should also be careful not to use a weapon which is inappropriate for the target; for instance, cruise missles against a submarine; nor to employ a weapon system beyond its tactical range. Also note that all commands are echoed prior to exiting from that section. If the player has made erroneous entries, he may enter Y(ES) to continue with another platform and reenter the platform's name over again. The only problem is that the vagaries of the programming will not allow the player to delete an employed weapon system. (All else may be changed.)

It should be noted that the coding is self-explanatory and user-friendly. By following the computer prompting, the players' only errors should be tactical!

(5) Damage suffered. The following example illustrates how platform self-damage is displayed just prior to accepting orders:

```
>boris

BORIS IS LOCATED AT POSITION 1522 SPEED 2


       NOTE
DUE TO PREVIOUS AND/OR CURRENT DAMAGE SUFFERED, MAX SPEED
    IS NOW 0
THE SHIP HAS BEEN HIT 3 TIMES


IT HAS BEEN HIT BY A TORPEDO
IT HAS BEEN HIT BY A NON-TORPEDO WEAPON
IT HAS BEEN HIT BY A TORPEDO

   THE BORIS    ATTACK ON TARGET LOCATED AT  720

RELEASE OF 21"          RESULTED IN
A DETONATION(HIT)

IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
   TYPE     LOCATION    METHOD    RANGE
THERE WERE NO DETECTIONS


SINCE THE BORIS  HAS BEEN SUNK, NO COMMAND INPUTS CAN BE
   ACCEPTED

ENTER (Y)ES TO CONTINUE WITH ANOTHER PLATFORM
ENTER (N)O TO RETURN TO DISPLAY SELECTION
```

When N(O) is entered, control is returned to the selection of display options.

E. GAME END

As previously discussed, there are three possible ways to end the game.

1. Game Turns

When the number of game turns exceeds the number input by the players, the following message is printed:

```
YOU HAVE JUST COMPLETED GAME TURN    2.
TYPE Y(ES) IF YOU WANT TO CONTINUE THE GAME, OR N(O)
TO ALLOW THE GAME TO END.
> n
```

If "NO" is entered, the game results are displayed.

2. Disengagement

When the number of destroyed ships for one player equals or exceeds the number he input for disengagement, the following statement is printed:

```
THE NUMBER OF BLUE SHIPS DESTROYED IS EQUAL TO OR
GREATER THAN THE NUMBER INPUT FOR GAME END.  IF YOU WANT
TO CONTINUE THE GAME, TYPE Y(ES).  TYPE N(O) TO END THE GAME.
>Y
```

3. Total Destruction

The total destruction of one player's ship forces will automatically end the game by displaying the final game results.

F. FINAL RESULTS

Game end results in the following information being printed:

```
BLUE RESULTS:
```

SHIP	STATUS/HITS	FINAL LOCATION
TINCAN	0	1102
ESCORT	SUNK	720
LAPON	0	714

```
TYPE Y(ES) TO SEE RED RESULTS.
>yes
```

RED RESULTS:

SHIP	STATUS/HITS	FINAL LOCATION
IVAN	SUNK	1311
BORIS	SUNK	1522

The players are also given the option of starting a new game:

```
TYPE Y(ES) TO RETURN TO MENU SELECTION AND START A NEW
GAME.  TYPE N(O) TO EXIT THE GAME.
> y
```

IV. PROGRAM DOCUMENTATION

A. INTRODUCTION

This chapter contains documentation for the approximately 2350 lines of FORTRAN source code which comprise the computer version of Up Scope. It is intended primarily for the reader who is interested in modifying the program. In using this chapter, the reader should refer to the source listing in Appendix F.

The program can be logically divided into the following parts:

1. Dimension, Common and Data Statements
2. Game Parameters and Weapon Systems Initialization
3. Game Control
4. Display Subroutines
5. Combat Execution Subroutines
6. Utility Subroutines

Each of the six parts will be described individually in the following pages. However, prior to a description of the FORTRAN source program, one should obtain an understanding of the major arrays and variables used in the program.

B. DESCRIPTION OF THE MAJOR ARRAYS

This section will only discuss the major arrays such as those used in the common statement and dimensioned in the main program for data arrays. All arrays are integer.

1. Arrays in common storage

a. PLATFM(100,37)

PLATFM is the largest and most important of all the arrays because it contains the basic information about each selected platform. Each platform has 37 columns associated with it. The size of the row dimension of PLATFM limits the game to 100 weapon systems. Aircraft, surface ships and submarines are included in the array. Due to the difference of their attributes, the columns of PLATFM associated with aircraft do not always serve the same purpose as those associated with ships, and only the first 24 columns of rows dealing with aircraft contain information. Any differences are discussed below.

(1) PLATFM(xxx,1). This column is used as a combat assignment flag. If the platform is an aircraft, a zero indicates no combat is scheduled, a one indicates combat is scheduled and weapon one is to be used and a two indicates weapon two is to be used. If the platform is a ship, a zero indicates no combat and a one indicates combat. This difference is due to the fact that aircraft can only fire one weapon per game turn while ships can fire up to three.

(2) PLATFM(xxx,2). This column holds the combat location which is the hex address to which the weapon/weapons will be delivered.

(3) PLATFM(xxx,3). This is the hex address of the current location of the platform.

(4) PLATFM(xxx,4). Current speed of the platform.

(5) PLATFM(xxx,5). This is the hex address of the future location of the platform.

(6) PLATFM(xxx,6). This column is used as a disturbed water flag. A zero means the platform is not within one hex of a disturbed water zone and a three means it is. A three is used because the routine that controls detections adds the contents of this column to the ship's evasion value.

(7) PLATFM(xxx,7). Column seven controls the search mission assigned to the platform. A zero indicates no search mission is scheduled, a one means the active sonor is to be used, a two means the passive sonor is to be used and a three means both are to be used.

(8) PLATFM(xxx,8). This column indicates the platform type. A zero means the platform is an aircraft, a one means a surface ship and a two means a submarine.

(9) PLATFM(xxx,9). This column is not used at present and is reserved for possible improvements to Up Scope.

(10) PLATFM(xxx,10). This column is not used at present and is reserved for possible improvements to Up Scope.

(11) PLATFM(xxx,11). For aircraft, this column contains a zero or a one. A zero means the aircraft is not

flying and a one means it is flying. For ships, the column contains the number of durability points lost.

(12) PLATFM(xxx,12). Max speed of the platform.

(13) PLATFM(xxx,13). Active detection value of the platform.

(14) PLATFM(xxx,14). Passive detection value of the platform.

(15) PLATFM(xxx,15). The number of type one weapons carried by the platform.

(16) PLATFM(xxx,16). The number of type two weapons carried by the platform.

(17) PLATFM(xxx,17). For aircraft, this is the row index of the type one weapon. For ships, this column holds the number of type three weapons carried by the platform.

(18) PLATFM(xxx,18). For aircraft, this is the row index of the type two weapon. For ships, it holds the evasion value.

(19) PLATFM(xxx,19). For aircraft, this column is not used. For ships, it is the defense value of the ship.

(20) PLATFM(xxx,20). For aircraft, this value is set at one. For ships, it contains the durability points of the ship. It will become zero when the ship is sunk.

(21) PLATFM(xxx,21). For aircraft, this column holds the row index number of the ship to which it belongs. For ships, it holds the index of the type one weapon.

(22) PLATFM(xxx,22). For aircraft, this is the row index for the type of aircraft. For ships, it holds the row index of the type two weapon.

(23) PLATFM(xxx,23). For aircraft, this column holds a number associated with the probability that weapon one hit its target. A zero means the weapon was not fired during the last game turn, a one through six means the number of chances out of six the weapon had of hitting its target, a seven means the weapon fired was a torpedo and there was a detonation and an eight means the weapon was a torpedo and there was no detonation. For ships, it holds the row index of the type three weapon.

(24) PLATFM(xxx,24). For aircraft, this column holds the same information as PLATFM(xxx,23) except for weapon two. For ships, it is a flag that indicates if simultaneous active and passive search missions are possible. A zero means simultaneous missions are not possible and a one means they are.

(25) PLATFM(xxx,25). This is a flag that contains a zero if the ship's passive sonor system suffers no degradation from speed and a one if degradation is caused by speed.

(26) PLATFM(xxx,26). The row index of the type one aircraft carried by the ship.

(27) PLATFM(xxx,27). The row index of the type two aircraft carried by the ship.

(28) PLATFM(xxx,28). The number of type one aircraft carried by the ship.

(29) PLATFM(xxx,29). The number of type two aircraft carried by the ship.

(30) PLATFM(xxx,30). The class row index of the ship. These numbers link the platform with the information found in Appendix B.

(31) PLATFM(xxx,31). The PLATFM row index number of the first type one aircraft carried by the ship.

(32) PLATFM(xxx,32). The PLATFM row index number of the first type two aircraft carried by the ship.

(33) PLATFM(xxx,33). The PLATFM row index number of the last type one aircraft carried by the ship.

(34) PLATFM(xxx,34). The PLATFM row index number of the last type two aircraft carried by the ship.

(35) PLATFM(xxx,35). This column holds a number associated with the probability that weapon one hit its target. A zero means the weapon was not fired during the last game turn, a one through six means the number of chances out of six the weapon had of hitting its target, a seven means the weapon fired was a torpedo and there was a detonation and an eight means the weapon was a torpedo and there was no detonation.

(36) PLATFM(xxx,36). The same information as PLATFM(xxx,35) except for weapon two.

(37) PLATFM(xxx,37). The same information as PLATFM(xxx,35) except for weapon three.

b. DETECT(100,3)

The DETECT array is used to store the detections that occur during a game turn. After each turn,the first column is initialized to minus one. If a game turn results in more than 100 detections a message is printed, but the game continues. All detections beyond 100 are lost.

(1) DETECT(xxx,1). Column one contains the PLATFM row index of the platform that is detected.

(2) DETECT(xxx,2). Column two contains the PLATFM row index of the platform that is detected.

(3) DETECT(xxx,3). Column three contains a flag indicating the type of sensor used. A zero means passive and a one means active.

(4) DETECT(xxx,4). Column four contains the distance in hexes between the detected and the detecting platforms.

c. WEP(29,3)

The WEP array contains the basic information about the 29 types of weapons portrayed by the game. The same information is contained in the data array WEPDA. The two are set equal so WEP can be commoned. The information contained in the array can be found in the Weapon Data Summary, Appendix C.

(1) WEP(xx,1). This holds the accuracy rating of the weapon.

(2) WEP(xx,2). This is the range of the weapon in hexes.

(3) WEP(xx,3). This is the index for the type of weapon. A one indicates a missile, a two a missile/depth bomb, a three a torpedo and a four indicates a depth charge.

d. HIT(100,3)

The HIT array is used to maintain the results of combat for a game turn. At the end of each turn the results are entered into PLATFM, and the first column of the HIT array is initialized to minus one.

(1) HIT(xxx,1). The PLATFM row index of the platform that fired the weapon.

(2) HIT(xxx,2). The row index of the type of weapon fired.

(3) HIT(xxx,3). The PLATFM row index of the platform hit by the weapon.

e. SUNK(100)

SUNK is the vector of PLATFM row index numbers of ships that have been sunk. This is a cumulative list in order of the occurrence of each sinking.

f. WATER(100)

WATER is a vector of the hex addresses that contain disturbed water as a result of a weapon detonating.

It is initialized to negative one prior to the combat execution phase.

g. NAME(100,6)

NAME is an alpha array that contains the user-supplied names of platforms. Each row is associated with the same numbered row in PLATFM. Its format is 6A1.

h. BUFFER(100,7)

BUFFER is the array used to hold the combat, move and search commands until the next game turn is started. This allows the user to recall each display option as many times as desired without disturbing the information contained in PLATFM. This information is input to PLATFM when the combat driver CONTRL is called.

(1) BUFFER(xxx,1). This column holds the combat assignment flag associated with PLATFM(xxx,1)

(2) BUFFER(xxx,2). This is the combat location desired for the next game turn.

(3) BUFFER(xxx,3). This is the speed desired for the next game turn.

(4) BUFFER(xxx,4). This is the desired location for the next game turn.

(5) BUFFER(xxx,5). This column assigns the type of search mission for the next game turn.

(6) BUFFER(xxx,6/7/8). These columns are used by non-aircraft to assign which weapons will be employed.

i. BLUPAS(6)

BLUPAS is the array that holds the blue player's alpha password. It is used before the initial loading of weapon systems and before each game turn to ensure the players cannot enter instructions to the other's platforms.

j. REDPAS(6)

REDPAS contains the red player's password and is used in the same way as BLUPAS.

3. Data Arrays

The data arrays (most of those used in the dimension statement) contain the basic information about the weapon systems portrayed by the game. Some of them are equivalenced to ease their entry and understanding. This will be discussed in the section dealing with the initial portion of the main program.

a. SHIPDA(130,18)

SHIPDA is the data array that holds the basic information about the ships portrayed by Up Scope. This information can be found in Appendix B. This information is transferred to PLATFM during the initial loading of the user selected weapon systems. Each column in SHIPDA is equivalenced to one of the arrays S1 through S18, and the data statements are actually in terms of the S arrays.

(1) SHIPDA(xxx,1)/S1(xxx). Max speed of the ship.

(2) SHIPDA(xxx,2)/S2(xxx). Active detection value of the ship.

(3) SHIPDA(xxx,3)/S3(xxx). Passive detection value of the ship.

(4) SHIPDA(xxx,4)/S4(xxx). A flag that is zero if active and passive sonor cannot be used simultaneously and a one if they can.

(5) SHIPDA(xxx,5)/S5(xxx). A flag that is zero if the passive sonor system capability is degraded by the speed of the platform and a one if no degradation occurs.

(6) SHIPDA(xxx,6)/S6(xxx). Evasion value of the ship.

(7) SHIPDA(xxx,7)/S7(xxx). Defense value of the ship.

(8) SHIPDA(xxx,8)/S8(xxx). The index of the type one weapon carried by the ship.

(9) SHIPDA(xxx,9)/S9(xxx). The number of type one weapons carried by the ship.

(10) SHIPDA(xxx,10)/S10(xxx). The index of the type two weapon carried by the ship.

(11) SHIPDA(xxx,11)/S11(xxx). The number of type two weapons carried by the ship.

(12) SHIPDA(xxx,12)/S12(xxx). The index of the type three weapon carried by the ship.

(13) SHIPDA(xxx,13)/S13(xxx). The number of type three weapons carried by the ship.

(14) SHIPDA(xxx,14)/S14(xxx). The index of the type one aircraft carried by the ship.

(15) SHIPDA(xxx,15)/S15(xxx). The number of type one aircraft carried by the ship.

(16) SHIPDA(xxx,16)/S16(xxx). The index of the type two aircraft carried by the ship.

(17) SHIPDA(xxx,17)/S17(xxx). The number of the type two aircraft carried by the ship.

(18) SHIPDA(xxx,18)/S18(xxx). Durability value of the ship.

b. NAMESH(130,8)

NAMESH is an alpha array that contains the class names of the ships found in SHIPDA and Appendix B. Each column of NAMESH is equivalenced to one of the arrays NS1 through NS8, and the data statements are actually in terms of the NS arrays. The format of NAMESH is 8A2.

c. ACDATA(16,7)

ACDATA is the data array that contains the aircraft information. The information it contains can be found in Appendix D.

(1) ACDATA(xx,1). Max speed of the aircraft.

(2) ACDATA(xx,2). Active detection value of the aircraft.

(3) ACDATA(xx,3). Passive detection value of the aircraft.

(4) ACDATA(xx,4). The index of the type one weapon carried by the aircraft.

(5) ACDATA(xx,5). The number of type one weapons carried by the aircraft.

(6) ACDATA(xx,6). The index of the type two weapon carried by the aircraft.

(7) ACDATA(xx,7). The number of the type two weapons carried by the aircraft.

d. NAMEAC(16,6)

NAMEAC is an alpha array that contains the names of the 16 aircraft. The format of NAMEAC is 6A2.

e. WEPDA(29,3)

WEPDA is the data array that contains the weapon data. It is equivalent to WEP which is discussed in the section dealing with the arrays in the common statement.

f. NAMEWP(29,6)

NAMEWP is an alpha array that contains the names of the weapons. Its format is 6A2.

g. INPUT(70)

This array is used to read in strings of alpha characters. All inputs are read in this way and then converted to the mode and format of the information actually required.

h. ALPHA(2)

ALPHA(1) contains a 'Y' and ALPHA(2) contains an 'N'. This array is used to compare user answers to questions that require a yes or no answer.

i. CHECK(6)

CHECK is used to compare user inputs to the Red or Blue passwords.

j. CLASNA(8)

CLASNA is used to read in the ship class names entered by the user.

C. DESCRIPTION OF MAJOR VARIABLES

The variables described in this section include only the major variables like those found in the COMMON statement. Other variables will be discussed as they appear in the program description. Normally single letter variables are used as local and temporary variables.

1. IPLAT

This variable is the total number of platforms loaded in the PLATFM array. It is used for loops that act on aircraft, surface ships and submarines.

2. IAIR

IAIR is the total number of aircraft loaded in the PLATFM array.

3. ISURF

ISURF is the total number of surface ships loaded in the PLATFM array.

4. ISUB

ISUB is the total number of submarines loaded in the PLATFM array.

5. ADV

ADV is the variable used to represent a ship's active detection value. It is used between all the detection subroutines.

6. PDV

PDV is the same as ADV for passive detection value.

7. AEV

AEV is similar to ADV and PDV except it represents a ship's active evasion value.

8. PEV

PEV is used the same way as AEV except for passive evasion values.

9. ERATIC

This variable is a flag that indicates the water condition of the playing area.

10. BLUAIR/REDAIR

These variables hold the number of Blue/Red aircraft entered by the players.

11. BLUSUR/REDSUR

These variables hold the number of Blue/Red surface ships entered by the players.

12. BLUSUB/REDSUB

The number of Blue/Red submarines entered.

13. PBAIR1/PBAIR2

All the variables of this form are pointers to the PLATFM array, and they are actually acronyms for what they represent. P(latform)B(lue)AIR1 is the row index number of the first Blue aircraft entered in PLATFM. P(latform)B(lue) AIR2 is the row index number of the last Blue aircraft entered in PLATFM. These variables are used in the initial loading of the PLATFM array and in the display subroutines. Figure 4 shows how the pointers are aligned, and how the data in PLATFM is arranged.

14. PRAIR1/PRAIR2

These variables are pointers to the first and last Red aircraft in PLATFM.

15. PBSUR1/PBSUR2

Pointers to the first and last Blue surface ships in PLATFM.

16. PRSUR1/PRSUR2

Pointers to the first and last Red surface ships in PLATFM.

17. PBSUB1/PBSUB2

Pointers to the first and last Blue submarines in PLATFM.

18. PRSUB1/PRSUB2

Pointers to the first and last Red submarines in PLATFM.

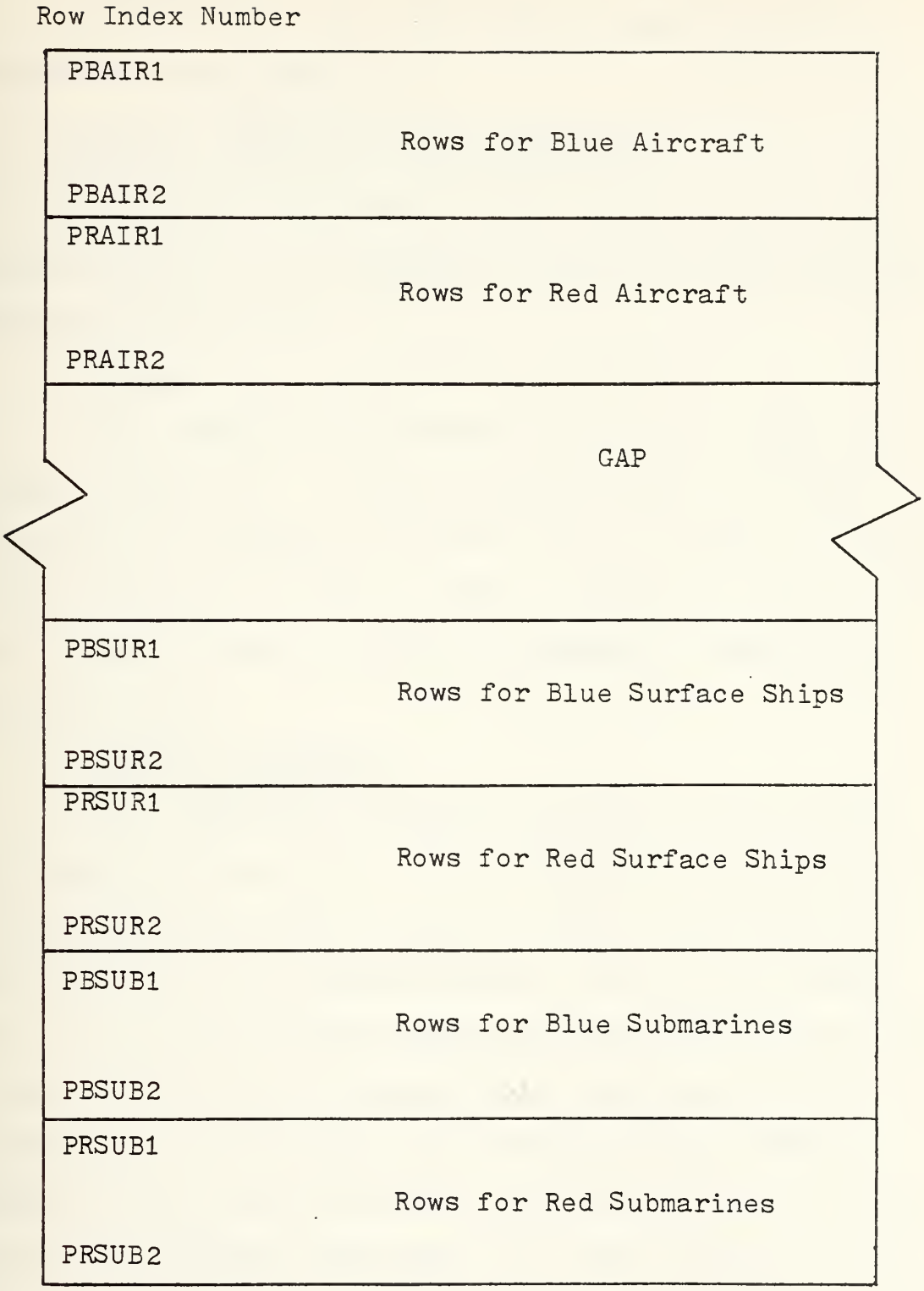

Figure 4. Format of the PLATFM Array

19. TURN

TURN stores the total number of game turns the players choose to play.

D. DIMENSION, COMMON AND DATA STATEMENTS

The first 194 statements of the program are dedicated to dimensioning the major arrays and setting up the data base required for Up Scope.

1. Line 1

All variables and arrays used in the program are integer except for: R which is used in the shot result subroutine to express the probability of a weapon hitting a target, RN which is used in FIRE to generate a random number and Y which is used in AEVSUB and PEVSUB in calculating evasion values.

2. Lines 2 through 17

The DIMENSION statement includes all the arrays discussed in section B.2. of this chapter. The majority of these arrays are used to ease the entry of data and are equivalenced to the arrays actually used by the program. The use of so many arrays eased the data entry because each array of the Sx type contained only one type of ship information. For example, S6 contains all the evasion values for the 130 ship types in Up Scope. If the evasion value of the America was wrong, the error would be in the second element of S6. It is in the second element because

the America is the second ship type in the data contained in Appendix B.

3. Lines 18 through 25

All the arrays and variables in the COMMON statement were discussed earlier in this chapter.

4. Lines 26 through 191

These lines are the actual data statements used to load the basic platform and weapon information. Each one of these arrays was discussed in section B.2. of this chapter. As previously mentioned, each Sx array is one column of the SHIPDA array, and each NSx array is one column of the NAMESH array.

5. Line 192

MAXPLT is used as the maximum number of platforms that can be played by the game. This limit is imposed by the size of the PLATFM and NAME arrays.

6. Line 193

ALPHA is an array used to check the player's answers to "yes" and "no" type questions.

E. GAME PARAMETERS AND WEAPON SYSTEMS INITIATION

Lines 194 through 742 are used to initialize all the arrays and variables, load the basic game parameters and to load the player-selected platforms.

1. Lines 194 through 201

This is a loop that sets the array WEP equal to the data array WEPDA so WEP can be placed in the COMMON statement.

2. Lines 202 through 211

These lines initialize the game parameter variables to zero.

3. Lines 212 through 231

These lines contain a nested loop that sets PLATFM, WATER, DETECT, HIT, SUNK and BUFFER to their initial values.

4. Lines 232 through 245

These lines print the format statement that gives the menu of choices and reads the player's choice. The alpha array INPUT is used to read the number of the player's choice as a character string, and the subroutine NONVRT discards any leading blanks and returns the integer form of the choice.

5. Lines 246 through 256

This group of statements will cause new arrays entered by the players to be reinitialized. It is an option that will allow the players to start a new game. Once the option is selected the players are asked if they want to continue and are given a chance to return to menu selection without erasing the arrays. The players answer to this question is read by INPUT, and any leading blanks and characters other than the first one are discarded by INVRT1. If the players want to restart the game, control is transferred to 99(line 205).

6. Line 257

This line transfers control to the STOP statement if the players indicate a desire to stop play and exit the game.

7. Lines 258 through 284

These statements print a warning about the maximum number of platforms allowed by the game, and read the number of game turns the game is to last and the water condition.

8. Lines 285 through 324

These lines read the basic game parameters and the password for the Blue side. The password is read with INPUT, and INVRT6 discards any leading blanks and any characters other than the first six. It returns the password as BLUPAS which is echo-printed to ensure the player knows his password as "understood" by the program. The variables BLUSUR, BLUSUB and BLUDES are also input and echoed by these lines with an opportunity to correct any errors.

9. Lines 325 through 364

Lines 325 through 364 perform for the Red player the same function lines 285 through 324 do for the Blue player.

10. Lines 365 through 375

These statements assign numbers to variables used as pointers to the PLATFM array during the initial loading of the platforms. Because the program automatically loads the aircraft carried by a ship and the combat, move and

search routines require aircraft to appear first in the array, surface ships and submarines are located at the bottom of the PLATFM array using these pointers. This method of loading leaves a gap between aircraft and ships unless there are exactly 100 platforms loaded. The gap, if it exists, is closed prior to the start of the game. Figure 4 as previously shown, shows the form of the PLATFM array and its pointers prior to closing the gap.

11. Lines 376 through 534

This section of the program performs the loading of the platforms selected by the Blue player.

a. Lines 379 through 397

These lines ask for, and read, the Blue player's password. If the reply is incorrect, he is given the opportunity to reenter the password or return to menu selection. DO LOOP 141 compares each element of the array CHECK with BLUPAS. A is set equal to one if any elements do not match and an error message is printed. If the entered password is correct, control is transferred to 8 (line 398) and the player proceeds with loading.

b. Lines 398 through 446

This section controls the loading of the Blue surface ships. SHIPNO is a local variable that is used to store the number of the Blue ships being loaded. It runs from zero to BLUSUR. Prior to entry into the loading loop, BLUSUR is checked against zero, and control is transferred

to the submarine loading section if no Blue surface ships were desired by the player. DO LOOP 165 is a long loop that does all the surface ship loading. It runs from PBSUR1 to PBSUR2 using SHIP as its index. The first steps of the loop are used to increment SHIPNO by one and print a format statement requesting a six character name for the SHIPNO ship. This player assigned name is stored in NAME(SHIP,x). This player assigned name is stored in NAME(SHIP,x). SHIP is also the row index number of PLATFM containing the information associated with NAME(SHIP,x). Next the player is asked to enter the ship class name of SHIP. The proper class names and the specifications for each class can be found in Appendix B. CLASNA is used to store the class name, and subroutine SCLASS is called to find the row index number of the class name in NAMESH that matches CLASNA. This index is stored in CLASIN, and it is associated with the row in SHIPDA that contains the basic information for that class of ship. CLASIN is stored in PLATFM(SHIP,30). The next piece of information entered is the initial hex location for the ship which is stored in PLATFM(SHIP,3). The last 19 statements of this section transfer the ship class information from SHIPDA and enter it in the appropriate columns of PLATFM. Sections B.1. and B.2. of this chapter discuss the contents of the columns of these arrays.

c. Lines 447 through 485

DO LOOP 155 is the loop that loads any aircraft SHIP carries. It runs from 26 to 27 because these are the columns of PLATFM that contain the aircraft index numbers. The first statement of the loop checks the column to see if an index number is entered. If the number is zero, control passes to the end of the loop. If the location contains an aircraft index number, the number is stored in ACINDX. The next statement prints the class name of the ship being loaded, the number of ACINDX type aircraft the ship carries and the name of the ACINDX type aircraft as stored in NAMEAC. Line 456 stores the row index number of the first ACINDX type aircraft in the parent ship's PLATFM row. Line 457 adds the number of ACINDX type aircraft carried by the ship to BLUAIR, and line 458 stores the row index number of the last ACINDX type aircraft in the parent ship's PLATFM row. DO LOOP 154 actually loads the aircraft information into PLATFM. Its index is AC, and it runs from P to Q which are set equal to the first and last PLATFM row index numbers of the aircraft. ACOUNT is the number of the aircraft being loaded. DO LOOP 154 also askes the player for a six character name for each aircraft. DO LOOP 153 enters the name in the NAME array. The next 12 lines transfer the required information from ACDATA to PLATFM.

d. Lines 486 through 534

This section loads the Blue submarines. The first statement checks BLUSUB to see if the Blue player entered a request for submarines. If not, control is transferred to the first statement of the Red loading section. DO LOOP 175 controls the loading, and it runs from PBSUB1 to PBSUB2. SUB is the index of the loop and SUBNO is the number of the submarine being loaded. The submarine loading statements follow the same pattern as the surface ship section except no aircraft are loaded. A six character name for the submarine is requested and entered in the NAME array. A class name and an initial hex location are also requested. The last 18 lines transfer data from SHIPDA to PLATFM.

12. Lines 535 through 692

These statements perform the loading of Red platforms, and they are almost an exact duplication of the Blue loading section. The only difference is in line 57 which sets REDAIR equal to BLUAIR. This is necessary to ensure the Red aircraft are properly loaded into PLATFM using the incremental process starting in line 615.

13. Lines 693 through 710

The first five lines of this section set the aircraft PLATFM pointers. The variables for each type of platform and total number of platforms are also set. The total number of platforms, IPLAT, is compared to 100. If it

is larger, the game must be restarted, and control is transferred to 99 (line 205).

14. Lines 711 through 742

These lines close the gap between aircraft and ships loaded into PLATFM. The first statement checks the location of the first Blue surface ship, PBSUR1, against the row index number of the last Red aircraft, PRAIR2, plus one. If these numbers are equal, there is no gap and the section is passed. If there is a gap, it is closed with a group of nexted do loops. DO LOOP 283 runs from one to ISHIP, the total number of Blue and Red ships, using I as an index. This loop moves through all rows of the PLATFM array that contain ships. DO LOOP 281 actually moves the information for each column of the selected rows. DO LOOP 282 closes the gap in the NAME array. DO LOOP's 284 and 285 check the ship's PLATFM row to see if it contains aircraft. If aircraft have been assigned, the PLATFM rows containing aircraft have new parent ship row index numbers placed in column 21 to adjust for closing the gap.

F. GAME CONTROL

Lines 743 through 877 of the main program control the actual "play" of the game turns. This is accomplished by several administrative functions and calls to subroutines. A simplified flow chart can be found in Figure 5. ITURN is the counter for the number of game turns played, and it

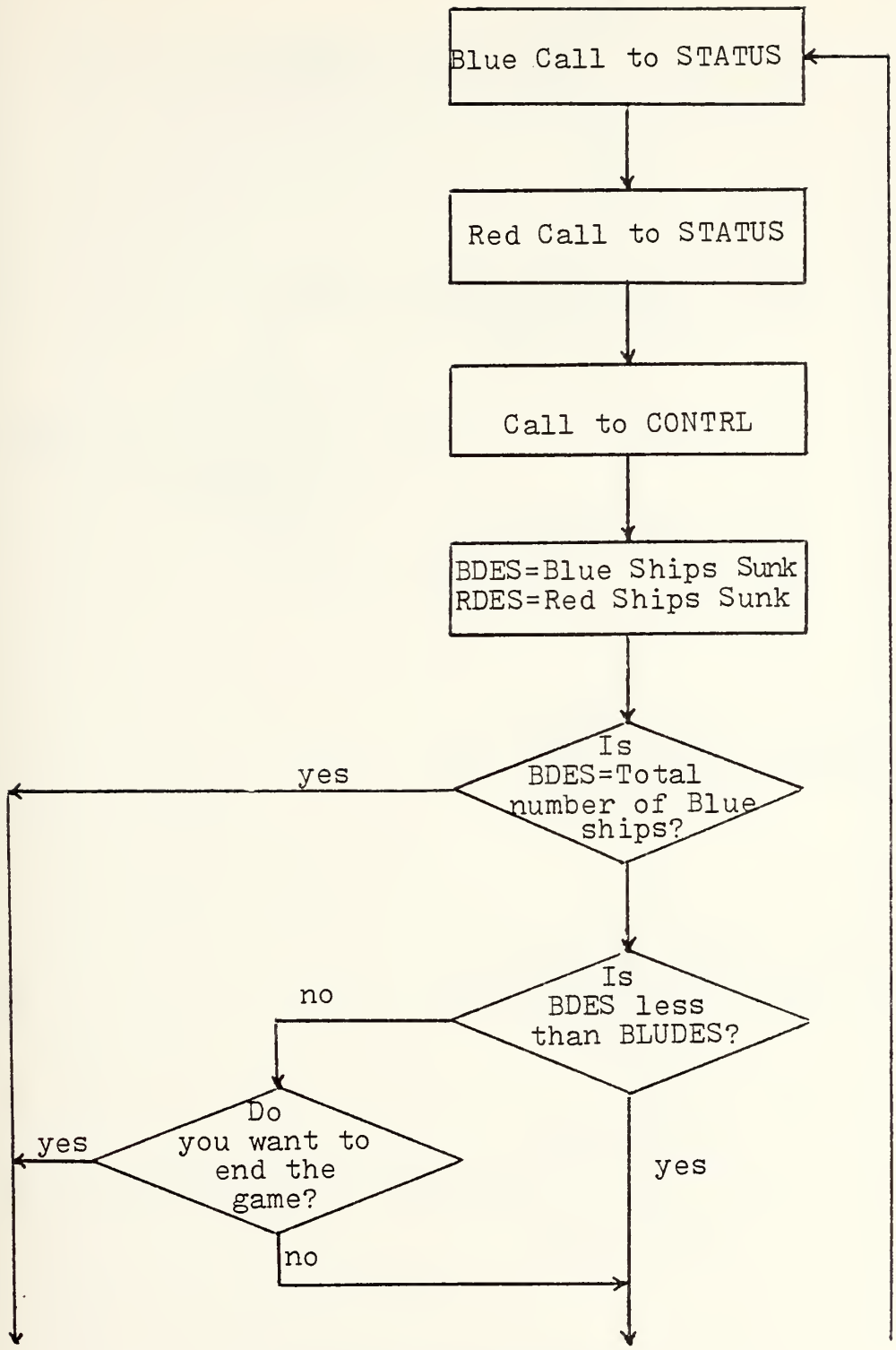

Figure 5. Simplified Game Flowchart

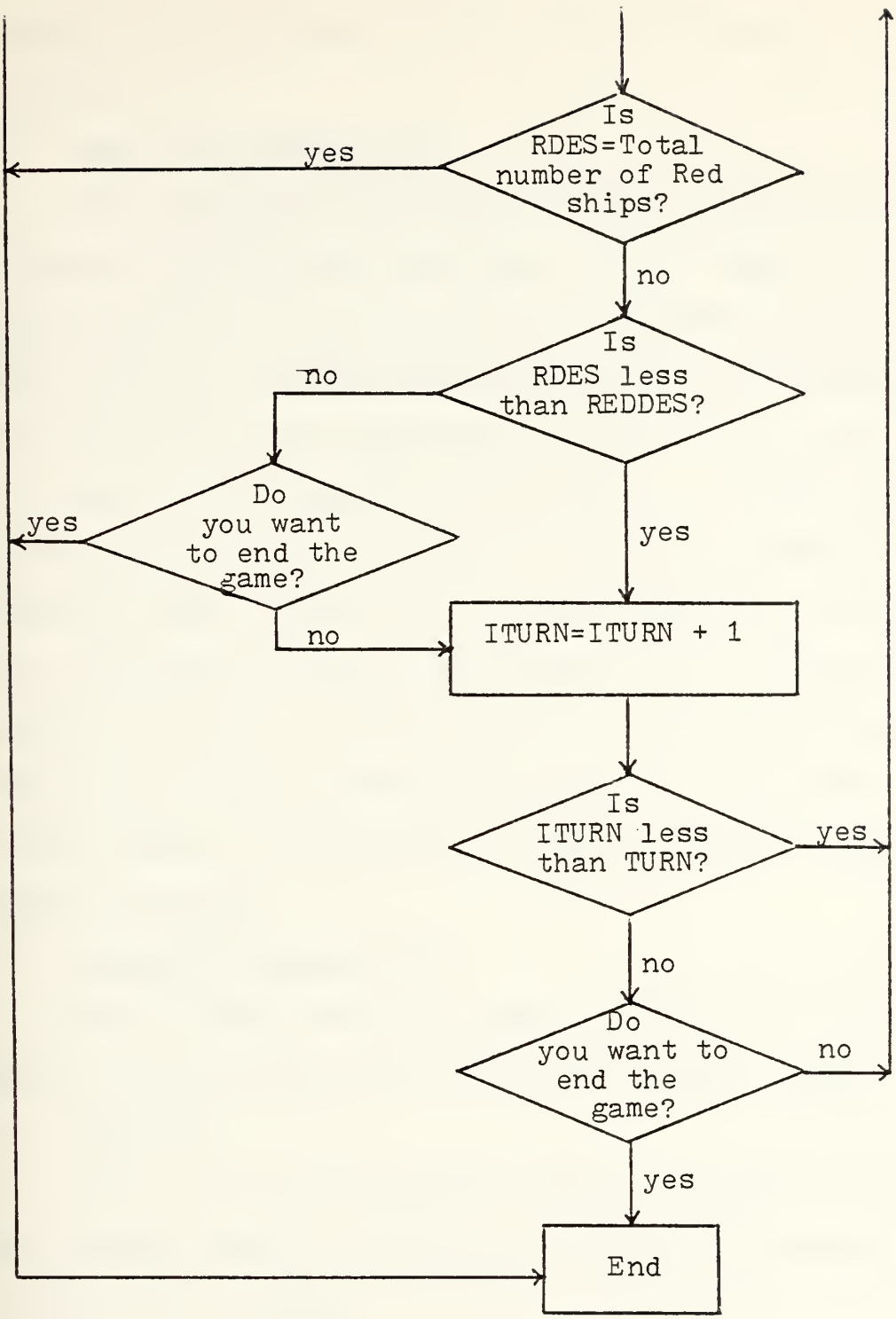

Figure 5 (continued)

is compared to the number stored in TURN to determine the normal game end.

1. Lines 747 through 759

The first action required for game control is to gain access to the display routines. This is done by requesting the Blue player to input his password and by calling subroutine STATUS discussed later in this chapter. STATUS will check the password and return an error code of zero in ERROR if the password is correct. If ERROR equals zero, STATUS will allow the player to view the results of the previous game turn and input instructions for the next turn. Control will then be transferred to 20 (line 760), the Red call to STATUS. If ERROR equals one, an error message is printed, and instructions are requested. The player can either reenter his password or return to menu selection to restart the game.

2. Lines 760 through 770

These lines perform the same function for the Red player as the previous section did for the Blue player.

3. Line 771

Line 771 calls subroutine CONTRL, discussed later in this chapter, which is the driver for all the combat, move and search routines.

4. Lines 772 through 781

BDES and RDES are local variables used to store the number of Blue and Red ships sunk after each game turn. DO

LOOP 303 loops through array SUNK, and increments the appropriate variable.

5. Lines 782 through 793

This section of statements asks several questions about BDES. If BDES is equal to the total number of Blue ships, the game is automatically ended by transferring control to line 815. If BDES is less than BLUDES, the player-selected number for premature game end, control is passed to 23 (line 792) which checks RDES. The last eight lines are reached only if BDES is less than the total number of Blue ships and greater than or equal to BLUDES. They print a statement informing the player of the situation, and give him the option of continuing or ending the game.

6. Lines 792 through 801

These lines check RDES for the Red player as lines 782 through 793 do for the Blue player.

7. Lines 802 through 811

Lines 802 through 811 increment ITURN and transfer control to the start of another game turn if it is less than TURN. If it is greater than or equal to TURN, the players are given the option of continuing or ending the game.

8. Lines 812 through 878

This section of statements ends the game by printing the status and final location of each ship. COUNT is a counter used to ensure the results are not pushed beyond the top of a CRT display. DO LOOP 313 prints the results of

the game. It runs from A to B with PLT as its index. A and B are variables set equal to the first and last PLATFM row index numbers of each of the four types of ships. C is a flag that indicates which of the four is being printed (one-Blue surface, two-Blue submarines, three-Red surface and four-Red submarines). Lines 833 and 834 increment COUNT and compare it to ten. If COUNT is less than ten, control is transferred to the CONTINUE statement of the loop. If COUNT is equal to ten, a new page of output is started. Lines 843 through 845 check the value of C, and transfer control to the appropriate section to increment A, B and C. For example, if C is equal to one, control passes all three statements, and goes to the Blue submarine section. There A is set equal to PBSUB1, B to PBSUB2 and C is set equal to two. Control then returns to the loop to print the Blue submarine results. If C is equal to two, control is transferred to line 863 which prints a message to begin the Red results. Then A, B, C and COUNT are updated for Red surface ships, and control is again transferred to the loop. After all four ship results have been printed, line 868 prints a message that gives the players a choice of starting a new game by returning to menu selection or letting the game end. If the game is allowed to end, line 874 prints an exit message.

G. DISPLAY SUBROUTINES

The display subroutines are called every game turn. They are used to display the results of the game turn and to

input combat, move and search instructions for the next turn. Four display options are available. They are detections of enemy ships, results of weapons fired, damage to friendly ships and individual display/input instructions.

1. Subroutine STATUS(PASWRD,ERROR,NAMEWP)

STATUS is the main driver for all the display routines. It is called with the player's password, PASWRD, and the array NAMEWP, and it returns an error flag in ERROR if the player entered the wrong password. No modification for Blue or Red calls is required because the subroutine assigns appropriate local variables based on the entered password.

a. Lines 13 through 30

DO LOOP 100 checks PASWRD against BLUPAS, and sets A equal to one if one of the elements does not match. If this is the case, control is transferred to 300 (line 31). If PASWRD does match, the calling player is Blue and lines 19 through 29 set local variables equal to the Blue PLATFM pointers and Blue game parameters.

b. Lines 31 through 47

These lines check PASWRD against REDPAS and if there is a match, perform the same operations as lines 14 through 29.

c. Lines 48 through 51

If neither BLUPAS or REDPAS match PASWRD, the player has made a mistake. These lines print an error message, and transfer control to the end of the subroutine with ERROR equal to one.

d. Lines 52 through 64

If either BLUPAS or REDPAS match PASWRD, control is passed to this section. The first seven lines print a format statement that asks the player to select and enter a display option. The answer is read by INPUT, converted to an integer by NONVRT and stored in A. Line 66 compares A to the possible option numbers, and prints an error message if it is incorrect. If A is a proper option, control is transferred to 830 (line 71).

e. Lines 71 through 82

These lines check A, and call the various display subroutines that correspond to the options. After each call, control is returned to 1000 (line 59), and the option menu is reprinted.

f. Lines 83 through 92

The last option is to start the next game turn. These lines print a format statement that asks the player if he has entered all combat, move and search instructions. If he does not answer yes, control is passed to line 59. Otherwise control is returned to the main program.

2. Subroutine DETXT(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP)

This subroutine loops through the DETECT array and prints the detections of the calling player's platforms. ALPHA1 is an alpha array that contains the word "SURFACE" in its first row and "SUBMARINE" in the second. ALPHA2 is an array that contains the words "PASSIVE" and "ACTIVE". COUNT is a counter for the number of lines the routine prints. When COUNT goes past a certain number, a new "page" of output is stated. This prevents detections from being pushed beyond the top of CRT display units.

DO LOOP 6 is the only loop in the subroutine, and it runs from one to 100 with PLT as its index. PLT is the number of a row of the DETECT array. D1 is set equal to the first column of DETECT which contains the PLATFM row index number of the platform doing the detecting. D2 is set equal to column two of DETECT which contains the row index number of the detected ship. If the first column of a row in the DETECT array is minus one, the previous row was the last detection, and control is transferred to the end of the routine. Lines 22 and 23 check the detecting platform to ensure it belongs to the calling player. If it does not, that row of the array is passed. If the detecting platform does belong to the calling side, line 25 prints four pieces of information. The first is the name of the detecting platform as stored in the D1 row of NAME. The second is the type of sonor used. A flag relating to the sonor type is

stored in the third column of DETECT. A one is added to this flag (zero-passive, one-active), and that row of ALPHA2 is printed. The third piece of information is the type of ship detected which is an index number stored in column eight of row D2 in PLATFM. This number yields the row of ALPHA1 which contains the ship type name. The last piece of information is the hex location of the D2 ship in PLATFM. Lines 28 and 29 increment COUNT and check it against the number 19. If COUNT is greater than or equal to 19, COUNT is set to zero, the heading is printed on a new page and control is passed back to DO LOOP 6. Otherwise control is maintained by the loop.

3. Subroutine SHOTRE(AC1,AC2,SUR1,SUR2,SUB1,SUB2, NAMEWP,SUR,SUB)

SHOTRE is the subroutine that displays the results of the weapons fired by the calling side. R is a real variable that stores the probability of a hit, and ALPHA1 is an array that holds the word "DETONATION" in row one and "NO DETONATION" in row two. COUNT is again used to count the number of lines printed so the results are not pushed beyond the top of a CRT display.

a. Lines 16 through 46

These lines print the results of weapons fired by aircraft. Line 16 transfers control to the ship loop if no aircraft are loaded. DO LOOP 10 runs from AC1 to AC2 with PLT as its index. Each number that PLT takes will be a

row index number of an aircraft in PLATFM of the calling side. Columns 23 and 24 of PLATFM rows containing aircraft hold the shot results for weapons one and two respectively. If both these columns are zero, neither weapon was fired, and line 21 will transfer control to the end of the loop. If one of the aircraft's weapons was fired, line 22 prints the name of the aircraft, the name of its parent ship and the combat location. DO LOOP 7 loops through both weapons carried by the aircraft to see which one was fired. Only one weapon can be fired by an aircraft per game turn. RESULT is set equal to the column which holds the results of the weapon firing. The numbers one through six represent the number of chances out of six that the weapon scored a hit. The number seven or eight is used when the weapon fired was a torpedo. A seven indicates the torpedo detonated (a hit) and an eight means it did not detonate (a miss). Line 30 prints the number of the weapon, the name of the weapon as stored in NAMEWP and the probability of a hit if the shot result is between one and six. If the result is seven or eight, line 34 prints the number and name of the weapon and "DETONATION" or "NO DETONATION". COUNT is then incremented and checked against five. If COUNT is greater than or equal to five, a new page of output is started and COUNT is set equal to zero.

b. Lines 47 through 69

These lines print the weapon result for ships, starting with the surface ships. A is set equal to SUR1, and B is set equal to SUR2. These are the PLATFM row index numbers of the first and last surface ships of the calling side. C is set equal to one as an indication that surface ships are being processed. DO LOOP 16 runs from A to B with PLT as its index. Line 57 checks the column that holds the shot results for ships, and passes control to the end of the loop if none of the weapons for the PLT ship have been fired. Next the name of the ship and its combat location are printed. DO LOOP 15 runs from weapon one to weapon three for each ship using WP as its index. It checks the results of the WP weapon, and prints a statement giving the probability of hit if RESULT is between one and six or a detonation result.

c. Lines 70 through 75

These lines increment COUNT, compare it to four and start a new page of output if COUNT is equal to four.

d. Lines 76 through 81

Line 76 checks the value of C, and passes control to the RETURN statement if it is greater than one. If C is equal to one, DO LOOP 16 has just processed the surface ship results, and must now do the submarines. C is set equal to two, A and B are set equal to SUB1 and SUB2, and control is transferred to the loop.

4. Subroutine DAMRE(SUR1,SUR2,SUB1,SUB2,SUR,SUB)

DAMRE prints the damage to the calling side's ships using the same double loop approach as SHOTRE.

a. Lines 18 through 32

DO LOOP 7 loops through the ships in PLATFM, and prints a "SUNK" statement if column 20 is equal to zero. If column 20 is not equal to zero, the ship is not sunk, and column 11 holds the number of hits. This number is then printed. Next COUNT is incremented, and a new page of output is started if it exceeds nine.

b. Lines 33 through 38

These lines check C, reset the variables A and B to SUB1 and SUB2 and return control to DO LOOP 7 if it is equal to one. Otherwise control is returned to STATUS.

5. Subroutine STATIN(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP)

This subroutine controls the individual platform displays/command acceptance subroutines which are divided between aircraft and surface ship/submarine sections. It is called by subroutine STATUS and calls STATAC, STATSS, NAMEIN, INVERT1 and INVERT6.

a. Lines 18 through 30

This section of code prints the heading telling the player that he is in the individual platform status section and to enter the name of the desired platform. It utilizes INVERT6 to preprocess the entry, then calls NAMEIN to convert the name of the desired platform to a row index.

If the name is either incorrect (spelling or otherwise) or belongs to the opposing player, a zero index is returned. If the index returned is zero the program prompts the player to reenter a valid name.

b. Lines 31 through 38

This section of code determines if the row index of the desired platform is an aircraft or a surface ship/ submarine and passes control to either STATAC or STATSS respectively. Again, there is a check against the other players platform being processed.

c. Lines 39 through 47

This section of code lets the player continue with another platform selection or return to the status menu selection.

6. Subroutine STATAC(USED,INDEX,INAME,NAMEWP)

This subroutine is all input/output related and displays a selected aircraft's individual status to include results of assigned combat missions, current speed, current location, results of search missions and damage inflicted on the aircraft. It accepts and processes all game turn command inputs for aircraft. USED is utilized by subroutine STATIN and is a flag set to indicate that STATAC has executed. INDEX is the PLATFM array row index of the desired aircraft. INAME is an array that contains the selected aircraft's name. NAMEWP is the array that contains all weapon system names.

Subroutine STATAC is called by subroutine STATIN and calls INVRT1, INVRT6 and DISTAN.

a. Lines 16 through 19

This section of code determines if the aircraft is currently flying. If not, control is transferred to 35 (line 99), thus bypassing prior game turn status.

b. Lines 25 and 26

Displays current location and speed.

c. Lines 27 through 39

This section of code displays results of combat action undertaken. It displays the target location and weapon employed. If weapon system one (two) was used then PLATFM column 23(24) will contain one to eight. If it is one thru six, then a non-torpedo weapon has detonated and the probability of a hit is that number out of six. A seven indicates a torpedo detonated while a eight indicates the torpedo missed and; therefore, did not detonate.

d. Lines 40 through 62

This section of code displays the search mission designated and the DO 20 loop identifies and displays all detections by the designated aircraft.

e. Lines 63 through 71

This section of code utilizes the weapon row index in PLATFM to display the available weapon system(s) name and maximum range.

f. Lines 77 through 92

This section of code accepts and processes an entered target location and weapon system number (one or two). This data is entered in the appropriate section of the BUFFER array. It then transfers control to 40 (line 107) bypassing the section that launches a non-flying aircraft.

g. Lines 99 through 101

Launches a non-flying aircraft by setting column 11 of PLATFM equal to one.

h. Lines 107 through 116

This section of code accepts and processes entered speed and future location inputs. This data is entered in the appropriate section of BUFFER.

i. Lines 122 through 128

This section of code calls DISTAN to determine the distance between future location and current location. It then checks to insure current maximum speed is not exceeded and that the distance computed is less than, or equal to, the speed command. If in error, control is transferred to 40 (line 107) and prompts the player to reenter speed and future location.

j. Lines 134 through 143

This section of code accepts and processes the search command. It insures that the aircraft has a speed command of zero. If not, it transfers to 50 (line 134) and the player restarts this section.

k. Lines 149 through 166. This section of code displays all entered and processed combat, move and search commands.

7. Subroutine STATSS(USED,INDEX,INAME,NAMEWP)

Subroutine STATSS is all input/output related and displays a selected surface ship/submarine individual status; to include results of assigned combat missions, current speed, current location, results of search missions and damage inflicted on the platform. It accepts and processes all game turn command inputs for surface ship/submarines. USED (line 18) is utilized by subroutine STATIN as a flag to indicate that STATSS has executed. INDEX is the PLATFM row index of the desired platform. INAME is an array that contains the selected platforms name. NAMEWP is the array that contains all weapon system names.

a. Lines 24 through 29

This section of code displays the platforms current position and speed, determines from PLATFM(x,11) how many times the platform has been hit and displays the number of hits received during the game.

b. Lines 35 through 42

This section of code searches the HIT array and displays hits (including type weapon), that the platform received during the previous game turn.

c. Lines 48 through 62

This section of code displays results of combat operations undertaken. It displays the target location,

then the DO 20 loop determines the weapon system(s) employed and displays employed weapon name and results of that attack in the same fashion that section IV.F.6.c does.

d. Lines 68 through 100

This section of code displays the results of the search mission(s). The DO 40 loop processes the DETECT array to display type platform, location, method and target range.

e. Lines 106 through 118

This section of code first determines if the platform has been sunk (during the previous game turn). If so, the subroutine bypasses the command input section and returns. The DO 51 loop displays weapon system(s) name(s) and range(s) available to the platform.

f. Lines 124 through 142

This section of code accepts and processes an entered target location and weapon system number for combat operations. This data is entered in the appropriate section of BUFFER. Column one of BUFFER is set to one if there is a combat mission assigned. Then column six, seven and eight are set to negative one depending on whether or not weapon system number one, two or three is employed. The program then determines if the player entered another weapon system number. If so, it transfers control to 60 (line 133) to accept another weapon system number.

g. Lines 148 through 169

This section of code accepts and processes entered speed and future location inputs for movement. It then calls a utility subroutine, DISTAN, to determine the distance between future location and current location and checks to insure current maximum speed is not exceeded and that the distance computer is less than, or equal to, the speed command. If incorrect, the program transfers to 70 (line 148) and restarts this section.

h. Lines 175 through 185

This section of code accepts and processes the search command. If both active and passive search is designated (three) and the flag to allow that is not set (PLATFM(x,24) equaling one) then control is transferred to 80 (line 175) to restart this section. The search command is then placed in the BUFFER array.

i. Lines 191 through 214

This section of code displays all entered and accepted combat, move and search commands.

H. COMBAT EXECUTION SUBROUTINES

The subroutines described in this section control and execute one complete game turn after the players have entered all their commands. The control, execution flow and layering of subroutines in this section are as follows:

	LEVEL		
1	2	3	4
CONTRL			
	COMBAT		
		FIRE	
			DISTUR
			RANGE
		FILHIT	
		DAMREC	
	MOVE		
	SEARCH		
		ADVSUB	
		AEVSUB	
		PDVSUB	
		PEVSUB	
		DETEK	

1. Subroutine CONTRL

Subroutine CONTRL controls the execution of combat, move and search phases for one game turn. It initializes all working arrays at the appropriate times. Subroutine CONTRL is called by the main program when all data display, command entry is complete and the players decide to execute the current game turn. It calls subroutines COMBAT, MOVE, SEARCH and utility subroutine DISTAN.

a. Lines 18 through 36

This section of code initializes the disturbed water array WATER and the hit array HIT in the DO 10 loop. In the DO 20 loop it removes previous combat operations and disturbed water zone entries. The DO 21 loop is executed for aircraft only and removes combat result information from PLATFM. The DO 22 loop removes combat result information from PLATFM for surface ships and submarines.

b. Lines 42 through 46

This section of code checks for non-flying aircraft and, for any, takes the future location of the parent ship from the BUFFER array and puts that in the aircraft's future location in BUFFER.

c. Lines 52 through 66

The DO 25 loop overlays the array BUFFER over selected columns of PLATFM. BUFFER contains the individual combat assignments, movement commands and search commands for the current game turn. Aircraft only carry one weapon system and therefore have no need to utilize columns six, seven and eight of BUFFER which activates the individual surface ship and submarine weapon systems.

d. Lines 68 and 69

Executes all combat and movement

e. Lines 75 through 78

The DO 30 loop initializes the detect array. This must be done after executing COMBAT and MOVE but before

executing SEARCH as SEARCH makes entries in DETECT. Subroutine COMBAT utilizes those entries.

f. Lines 83 through 95

This section of code compares all current platform locations with the locations of disturbed water zones in the WATER array. The DO 40 loop sets NUM equal to the number of disturbed water locations. The DO 70 loop increments through all platforms bypassing ones that are either sunk or have had a disturbed water zone recorded previously in the current game turn. The DO 60 loop determines the distance from a platform to every disturbed water location and, if the distance is zero or one, places a disturbed water marker, a three, in column six of the array PLATFM.

g. Line 101

Executes the search phase by calling SEARCH.

h. Lines 107 through 119

This section of code reinitializes the BUFFER array prior to exiting the execution stage of the game turn and prior to accepting future entries. BUFFER(x,4), future location, is set equal to the current position in case a player does not issue a move command for that platform.

2. Subroutine COMBAT

This subroutine executes all phases of air/surface to subsurface, and subsurface to surface/subsurface combat for all platforms. It is called by CONTRL and calls FIRE, FILHIT and DAMREC.

a. Lines 14 through 17

The DO 30 loop extends to line 48 and executes the basic combat assignments for all platforms excluding ones that are sunk, and ones that do not have combat assignments. Line 17 directs control to either aircraft or surface ship/submarine combat execution.

b. Lines 23 through 31

This section of code executes a combat assignment for an aircraft. WEPIND is the column in PLATFM that contains the weapon system row index (for the weapon data array WEP) of the weapon system to be employed. WEPSUP is the column in PLATFM that contains the weapon resupply value for the employed weapon system. Subroutine FIRE is called, passing ATTK(row index of attacking platform), WEPIND, and WEPSUP. IHIT (zero or one, a one if the attack resulted in a hit), ITAR (row index of the target platform) and PRHIT (probability of obtaining a hit) is returned by FIRE. PRHIT is expressed as an integer between one and six. X is the weapon row index. Lines 27, 28 and 29 place the combat results into column 23 or 24. If the attack was successful, subroutine FILHIT is called passing the attacking platforms row index, weapon system row index and target platforms row index in that order. FILHIT records the hit. It then transfers control to the damage assessment section.

c. Lines 37 through 48

This section of code executes a combat assignment for a surface ship/submarine. WEPIND and WEPSUP are as

described above in the aircraft section. The DO 20 loop searches the weapon system row index columns looking for a negative row index which signifies a combat mission for that weapon system. When encountered, the negative sign is removed and subroutine FIRE is called. As previously stated above, ATTK, WEPIND, and WEPSUP are passed and IHIT, ITAR and PRHIT are returned. The combat results are entered in the PLATFM array and subroutine FILHIT is called if the attack was successful.

d. Lines 54 through 59

The DO 40 loop calls subroutine DAMREC to process the damage resulting from the game turn. It passes the row index of the hit platform.

e. Lines 66 through 79

The disturbed water array WATER contains the locations of all detonations that occurred during the current game turn and may contain multiple locations from multiple attacks on a target. This section of code removes all duplicate locations and closes up the array WATER.

3. Subroutine FIRE(ATTK,WEPIND,WPLEFT,IHIT,ITAR,PRHIT)

Subroutine FIRE executes a weapon firing, computes weapon and target range and calculates a random number to compare against a computed probability of a hit to determine hit or miss. It also determines disturbed water locations. ATTK is the attacking platform's row index. WEPIND is the employed weapon row index. WPLEFT is the weapon resupply

value. IHIT is one or zero, a one signifying a hit was obtained. ITAR is the targets row index. PRHIT is the computed probability of a hit. Subroutine FIRE is called by subroutine COMBAT and calls subroutine DISTUR, RANGE, DISTAN and RANDOM.

a. Line 18

Immediately returns to COMBAT if the associated weapon resupply is zero.

b. Lines 24 through 38

This section of code decrements the associated weapon resupply by one. The DO 10 loop sets ITAR equal to the row index of a platform whose current location matches the attackers combat location. If there is not a match, the subroutine returns.

c. Lines 37 and 38

Subroutine DISTUR is called to place a disturbed water marker into WATER using the target's future location.

d. Lines 44 through 49

The DO 15 loop sets Y equal to one if the attacker previously detected the target. If not, the subroutine returns.

e. Lines 55 through 59

This section of code insures that only a weapon capable of attacking that particular type of target is employed, if not, the subroutine returns. It then checks to insure that air/surface platforms only attack subsurface

platforms and that subsurface platforms only attack surface or subsurface platforms, otherwise the subroutine returns.

f. Lines 68 through 75

This section of code determines target range (DIST) from DISTAN, weapon accuracy (ACC) and weapon range (RANG) from RANGE. If target range exceeds weapon range, control is transferred to 40 (line 84) to inform the players of that fact. It then calculates the probability of a hit by subtracting the targets defense value (PLATFM(ITAR,19)) from the weapon accuracy (ACC). By definition, if ACC equals three, the probability of a hit is one out of six. The program calls RANDOM and generates an integer random number from one to six. If PRHIT is greater than or equal to the random number, a hit occurs(IHIT equal to one).

g. Line 82

DISTUR is called to place a disturbed water marker at the targets future location if the weapon was a torpedo and a hit.

4. Subroutine DISTUR(LOCAT)

Subroutine DISTUR utilizes LOCAT which is the hex address of a weapon that has detonated and caused a disturbed water zone. It files it in the disturbed water array WATER. It is called by subroutine FIRE and does not call any subroutines.

a. Lines 13 through 15

This section of code finds the next unused row of the array WATER.

b. Lines 16 through 18

This section of code tells the player there has been over 100 detonations, therefore the players cannot be certain which detonations will cause disturbed water zones. It then returns to subroutine FIRE.

c. Line 19

If there is room in WATER, the location of the disturbed water is entered.

5. Subroutine RANGE(INDEX,ACC,RANG)

Subroutine RANGE utilizes weapon system row index (INDEX) to enter the Weapon Data Summary array (WEP) and pass back the weapon system accuracy (ACC) and the weapon system maximum range (RANG). Subroutine RANGE is called by subroutine FIRE and does not call any subroutines.

6. Subroutine FILHIT(ATTK,WEPON,ITAR)

Subroutine FILHIT files all 'hits' on surface ships and submarines as they occur. FILHIT is called by the subroutine COMBAT and does not call any subroutines.

a. Lines 13 through 15

This section of code locates the next unused row of array HIT.

b. Lines 16 through 18

This section of code tells the players when over 100 hits have occured. It should be noted that if this happens, only the weapon result displays to the players are affected and that weapon results for some firing will not be available. The actual damage assessment and capability reduction is conducted regardless.

c. Lines 19 through 21

This section of code enters the target's row index, weapon system row index and attacking platform's row index in that order.

7. Subroutine DAMREC(ITAR)

Subroutine DAMREC processes damage inflicted by a hit into the PLATFM array, and places ships that are sunk in the SUNK array. It is called by subroutine COMBAT and does not call any subroutines.

a. Lines 18 through 22

This section of code decrements the following by one:

Column of PLATFM	Identity
12	maximum speed
13	ADV
14	PDV
15	weapon one resupply
16	weapon two resupply
17	weapon three resupply

Column 11 of PLATFM contains durability point losses suffered. For ease in handling internally, it is carried as a negative value.

b. Lines 23 through 35

This section of code checks to see if the platform is still 'alive' and ensures it had not been sunk previously during the current game turn. If so, the program returns control to subroutine COMBAT. If the platform was sunk during this iteration, it locates the next available row in the SUNK array.

c. Lines 37 and 38

This section of code enters the row index of the sunk platform into SUNK and sets the durability value of the platform equal to zero (PLATFM(x,20)). This is the general indicator for the rest of the program. That is, if column 20 of PLATFM is zero for any platform, the platform is ignored for all practical purposes.

8. Subroutine MOVE

Subroutine MOVE executes all platform movement except platforms that have been sunk. It is called by subroutine CONTRL and does not call any subroutines.

a. Lines 14 through 17

This section of code replaces the current location (PLATFM(x,3)) with future location (PLATFM(x,5)). The DO 10 loop skips platforms that have been sunk.

9. Subroutine SEARCH

Subroutine SEARCH executes all phases of active and passive search missions after combat operations and movement have been executed. Note: Surface units cannot search against other surface units and aircraft cannot detect surface ships. It is called by subroutine CONTRL and call subroutines ADVSUB, AEVSUB, PDVSUB, PEVSUB and DETEK.

a. Lines 23 through 36

This section of code executes surface and air active search missions against subsurface platforms. If there are no surface or air platforms involved in the game, the program transfers to 45 (line 62). The DO 20 loop executes for aircraft and surface platforms with an active mission that are not sunk. The program calls subroutine ADVSUB to calculate the active detection value (ADV) for the searching platform. Then the DO 10 loop processes all submarines, bypassing those sunk, and calls subroutine AEVSUB to calculate the active evasion value (AEV) and range (DIST) between the two platforms. If the ADV is greater than or equal to the AEV, there is a detection and subroutine DETEK is called to file the detection in the DETECT array. The row indices of searcher and target, active/passive designation and range are passed.

b. Lines 43 through 55

This section of code is almost identical to the section described above except that it is the passive

search mission that is being executed. The passive detection value (PDV) and passive evasion value (PEV) are substituted for their active counterparts.

c. Lines 61 through 76

This section of code executes submarine active search missions against both surface ship and submarine platforms. It is coded in the same fashion as above except that the limits of the DO loops and indices are changed to reflect the differing platforms.

d. Lines 83 through 97

This section of code executes submarine passive search missions against surface ship and submarine platforms. It is coded in the same fashion as above except that the limits on the DO loops and indices have changed and it is executing passive vice active searches.

10. Subroutine ADVSUB(I)

This subroutine computes the active detection value (ADV) for a platform executing an active search mission. It utilizes I as an index to access the proper row of PLATFM. It is called by subroutine SEARCH and does not call any subroutines.

a. Line 15

Sets ADV equal to the active detection value for that platform.

b. Line 16 through 19

This section of code bypasses adjustments for damage and speed if the searching platform is an aircraft.

The ADV is decremented for both damage suffered and current speed of the platform.

c. Lines 20 through 22

This section of code decrements ADV for both erratic water conditions (ERATIC equaling one) and the platform being within one hex of a disturbed water zone (PLATFM(x,6) set to three). The ADV must be nonnegative.

11. Subroutine AEVSUB(I,J,DIST)

This subroutine calculates the active evasion value (AEV) for a potential target platform. I is the row index of the searching platform and J is the row index of the potential target. DIST is the distance between platforms I and J. It is called by SEARCH and calls DISTAN.

a. Lines 17 through 21

This section of code determines the distance between platforms I and J, then calculates the AEV by dividing the distance by three and rounding up. Note that Y is a real and the AEV is one even if the distance is zero. The resulting AEV is then increased if the target platform is in a disturbed water zone (line 21).

12. Subroutine PDVSUB(I)

This subroutine computes the passive detection value (PDV) for a platform executing a passive search mission. It utilizes I as an index to access the proper platform in PLATFM. It is called by subroutine SEARCH and does not call any subroutines.

a. Line 15

Sets PDV equal to the PDV value in PLATFM.

b. Lines 16 through 20

This section of code makes PDV adjustments for damage and current speed. If the platform is an aircraft (line 16) this section is bypassed. Also, if the platform does not suffer degradation of capability due to speed (line 18), this section is bypassed.

c. Lines 21 through 23

This section of code makes PDV adjustments for erratic water conditions (ERATIC equaling one) and for being within one hex of a disturbed water zone (PLATFM(x,6) is set to three). The PDV must be nonnegative.

13. Subroutine PEVSUB(I,J,DIST)

This subroutine calculates the passive evasion value (PEV) for a potential target platform. I is the row index of the searching platform and J is the row index of the potential target platform. DIST is the distance between platforms I and J. It is called by subroutine SEARCH and calls DISTAN.

a. Lines 17 through 21

This section of code determines the distance (DIST) between platforms I and J then calculates PEV by dividing by ten and rounding up. Note that Y is a real and the PEV is set to one even if the distance is zero. This PEV is then added to the evasion value (PLATFM(x,18)) and

and adjusted for being within one hex of a disturbed water zone.

b. Lines 22 through 24

This section of code adjusts the PEV downward if the platform executed a combat mission and/or was assigned an active search mission. The PEV must be nonnegative. If the target platform has a speed of zero, did not execute a combat mission and was not assigned an active search mission the PEV is defined to be six.

14. Subroutine DETEK(II,J,AP,DIST)

Subroutine DETEK files all detections in the detection array DETECT as they occur. II is the row index of the searching platform, J is the row index of the detected platform, AP is one if the detection was by active means and a zero if by passive means, and DIST is the distance between the platforms II and J. It is called by subroutine SEARCH and does not call any subroutines.

a. Lines 14 through 16

This section of code locates the next unused row of the array DETECT.

b. Lines 17 through 19

This section of code tells the players when over 100 detections have occurred, that detection and all future detections will not be processed as detections.

c. Lines 20 through 22

This section of code places the row index of the search platform in column one, the row index of the detected platform in column two, the active/passive indicator in column three and the distance in column four of the DETECT array.

I. UTILITY SUBROUTINES

The subroutines described in this section perform functions that can be considered "administrative" and not related to any particular part of the program previously discussed.

1. Subroutine NONVRT(INPUT,OUTPUT)

This subroutine takes INPUT which is a character string of numbers (70A1), discards any leading blanks and returns the integer form of the string as OUTPUT. It is used by the program to ease the entry of numbers that would otherwise require restrictive format statements. All variables in the subroutine are integer mode and three integer arrays are used. NUMBER(10) is a data array that contains the ten primary digits, zero through nine. X(70) is an array used for the temporary storage of the numbers in INPUT. BLANK and DOT are alpha variables that relate to a blank space and a decimal point. DO LOOP 1 is a loop that checks each column of INPUT for blank spaces. The first column number that contains a character is stored in IFNUM. DO LOOP 3 checks the columns IFNUM through 70 of INPUT for

the blank or decimal point that ends the character string. ILNUM is set equal to this column number minus one. IFNUM and ILNUM are the inclusive column numbers of INPUT that contain information, and INUM is the total number of columns that contain information. DO LOOP 6 runs from one to INUM with I as its index. DO LOOP 5 is nested in DO LOOP 6, and it compares each column of INPUT to the character digits in NUMBER. The number of the character that matches is placed in X(I). DO LOOP 7 runs from one to INUM with J as its index, and it actually computes the number placed in OUTPUT. INDEX is the number of the column of X being worked on, and it is decremented from right to left. OUTPUT is calculated by multipliying X(INDEX) by a factor of ten raised to the J minus one power. For example, the far right column would be multiplied by ten to the zero power, and added to OUTPUT. The third column from the right would be multiplied by ten to the second power or 100, and added to OUTPUT.

2. Subroutine INVRT1(INPUT,IPUT) and INVRT6(INPUT,IPUT)

These subroutines perform the same basic function which is to read a character string and return it without leading blanks. INVRT1 returns only the first character, and is used to record Y(es) and N(o) answers as "Y" and "N". It does this with one loop that assigns the first column of INPUT not equal to a blank space to IPUT. INVRT6 discards any leading blanks and returns the first six characters of a string. It is used for recording platform names. The

subroutine uses a nested loop with the outside loop searching for the first column of INPUT to contain information. The inside loop moves the first six columns of INPUT containing information to IPUT, and transfers control to the RETURN statement.

3. Subroutines SCLASS(NAME,INDEX,NAMEWP) and NAMEIN (INPUT,INDEX,NAME,AC1,AC2,SUR1,SUR2,SUB1,SUB2)

These subroutines compare alpha names to the names in arrays, and return the row index numbers of the names that match. SCLASS searches the 130 class names in NAMESH with a simple search that just compares the first four columns of the arrays. If none of the names match, INDEX is returned as zero. NAMEIN searches the user supplied names located in the array NAME. This search is more complicated because the row index numbers are checked to see if they match the rows in PLATFM controlled by the calling player. This prevents the wrong index number being returned if both Blue and Red players use the same name for a platform.

4. Subroutine DISTAN(XX,YY,D)

Subroutine DISTAN computes and returns a hex distance (D) between any two hex locations that are passed to it (XX,YY). It is called by subroutine STATAC, STATSS, CONTRL, FIRE, AEVSUB and PEVSUB. It does not call any subroutines.

a. Lines 14 through 18

This section of code separates the two hex locations XX and YY into their two components, the column

component the first two digits and the row component composed of the last two digits. It should be noted that this procedure is what limits the size of the game board. If the 100's in lines 14 through 17 were changed to 1000's, then the upper limit on the game board and in this program would nominally be 999 by 999. Actually, however, only the row index is limited to two digits. The column index is unlimited except for the hardware limit of the computer being utilized. D is initialized to zero and is the distance.

b. Lines 19 through 38

This section of code first compares the columns of the two locations. If they are equal then the distance is the absolute value of the difference in the row coordinates. If the column coordinates do not match, then through a series of comparisons, the row and column of the XX location is incremented or decremented to make it appear that the XX location is taking a 'step' closer to the YY location.

c. Lines 39 through 41

After the 'step' is taken above, the distance is incremented by one and the locations are compared column to column and row to row. If there is no difference in the coordinates in either location the subroutine returns the current value of D. Otherwise it transfers control to 100 (line 19) and section b above is repeated.

V. METHODOLOGY

A. INTRODUCTION

This chapter outlines a general structure that eases and provides a framework for converting tactical board games to computer-assisted war games. The structure presented is based on lessons learned during the conversion of "UP SCOPE!." It should be remembered that the program presented in Chapter IV is not the result of the outline described but just the opposite. Although the ideas presented here were largely incorporated in the program, the outline itself was not available during programming. The experience gained during this effort would result in some changes to the program if the programming effort was started again.

An effort to convert a board game to a computerized game must take into consideration the following five major areas:

1. The amount of computer assistance.
2. The rules of the game.
3. The procedures required by the game.
4. The structure of the data.
5. The input and output of information.

Although each of the above areas is related; they should, in general, be considered separately in the specific order presented.

B. COMPUTER ASSISTANCE

Computer assistance can be defined as the degree to which the computer performs the tasks previously required of the players. The assistance can range from just the calculation of hit probabilities all the way to graphic displays and voice input. The amount of computer assistance supplied by the game is dependent on the following three factors:

1. The Programmer

The first factor involves the programmer. His ability and ideas concerning a computerized game will have a large effect on how much of the game he decides to program. However, this decision is usually made by the programmer's superiors.

2. The Budget

One resource restriction involves the amount of money available for the effect. In this sense "money" includes equipment, as having money implies the ability to spend it on equipment. Budget considerations can largely be ignored by the programmer because he is given the parameters within which the assigned tasks must be accomplished. However, the manager must be aware of the tradeoffs between money and the computing capacity and speed of the computer, and how they will effect the amount of assistance the program can provide.

3. Time

The time factor involved is another resource restriction that can effect the amount of assistance provided by the program. The time factor can be driven by either the programmer or the player. Usually the degree of assistance is designed to minimize the player's time and not the programmer's, because a successful game will be played many times. An awkward game, which may have minimized the programmer's time, will waste more of the organization's personnel hours by forcing the players to play longer than need be.

4. Result

The assistance provided by the computerized version of Up Scope in relation to the board game is almost total. The only task required of the players, other than to be aware of the rules and information contained in the data summaries, is to maintain the game board with ship locations.

C. RULES AND PROCEDURES

The rules of a board game and its procedures are closely related. The difference lies in the idea that rules imply "can" or "cannot" and therefore "should" be adhered to in order to play the game correctly. Rules also usually involve some choice on the part of the players. They can be broken and yet the game can go on. Procedures on the other hand are the skeleton of the game and involve no choice on

the part of the players. They give the game its framework around which the players make their decisions. An example of a rule in Up Scope is limiting the number of hexes a ship can move. A procedure would be the calculation of a hit probability and the rolling of the die to determine the hit. In a conversion effort the rules and procedures become computer code and instructions. The programmer must become extremely knowledgeable about them because his task is to translate them to computer code with no accidental changes.

Changes in the rules and procedures can be the result of two things. The first is the fact that the programmer is restricted in some way that necessitates a change. The second is to improve the game in some fashion. This normally means a change to make the game more realistic. Prior to coding the first computer statement, detailed flow charts should be prepared for each aspect of the game. These flowcharts will force the programmer to study and learn the rules and procedures of the game. They will also ensure the program is a "faithful" reproduction of the board game, or highlight any changes that have been made.

Caution when dealing with rules and procedures is required. Although they are closely related, rules involve interpretation and choice. It should be remembered that the conversion of a game to a computer program in effect makes rules like procedures in that the player is confined to acting in a certain manner. The game becomes more rigid and

self-umpiring. The purpose of programming a board game is to free the players from bookkeeping and computational chores but in the process it adds restrictions. The programmer should be aware of this tradeoff, and guard against an overly restrictive game.

The following are some of the changes made to Up Scope:

1. Weapons

SPI's procedure for handling the resupply of weapons involves a resupply number. This number is used to limit the number of weapons each platform can carry and fire. Each weapon can be fired once without a change in procedure, but after the first shot a qualifying roll of the die must be made to determine if the weapon can be fired again. If the resupply number is higher than the die, the weapon can be fired. If not, the weapon cannot be used until the next game turn when the procedure will be repeated. Every time the weapon is fired, its resupply number is reduced by one. The computer version of "UP SCOPE!" does not make the qualifying die roll described above because the procedure becomes too restrictive and unrealistic when the resupply number becomes small.

2. Hits

Another change designed to increase realism is the amount of information supplied to the players about weapon results. The SPI procedure gives the offensive player perfect information about each shot result. The

computerized version gives the player the probability of a hit or a detonation result if the weapon is a torpedo.

3. Aircraft

The main change concerning aircraft is the max speed of some of the aircraft. They were changed to reflect current capabilities.

D. DATA STRUCTURE

Once the amount of computer assistance has been determined and the flowcharts have been set, the data structure for the game can be finalized. Of course some consideration must be given to the data structure while the above is being accomplished, but it is critical that the data structure be finalized in its chronological place. The order should have the flowcharts prior to entering the first line of code. Too often coding is started prematurely in an attempt to get "something on paper". The proper sequence forces the programmer to consider all the game's data requirements before he commits himself to actual code. Any other procedure will result in numerous changes and additions.

Usually the data required by the game takes one of two forms. The first is basic data which includes game parameters, system specifications and assumptions. The basic data for Up Scope includes the data summaries in Appendices B, C and D. Information presented by a scenario can also be considered basic data. In general basic data

includes all the information that is available to the players. The second form of data consists of the working files used by the program to accomplish its tasks. The working files will be transparent to the players.

1. Basic Data

Basic data is often not a primary concern of the programmer because unlike the working files it is not manipulated or used extensively by the program. This should not be the case because it is possible to improve the program by careful structuring of the basic data. In the computer version of Up Scope, the basic data is entered using data statements. As previously mentioned, the ship data is entered using 18 different data arrays, one array for each separate piece of information. All of these arrays are equivalenced to SHIPDA which is used in the platform loading. Although this seems like a great deal of effort when one or two large data arrays could be used, it saved a great deal of time finding and correcting errors in coding. This was accomplished with no increase incomputing time or storage requirements. Another aspect of the basic data structure which improved the program was the ordering of the countries' ships in SHIPDA. The order in SPI's rule book has the USSR first and the United States fifth. By itself this seems unimportant, but several times per game turn a search of the arrays to find a specific ship class is required. Under the assumption that players will use the

United States' and USSR's ships more often, the search times were reduced by placing the ships in that order.

2. Working Files

The working files are of major concern to the programmer because they provide a focus for his efforts. This is true in the sense that all the actions of the program are directed towards accepting, manipulating, retrieving and providing information. The rules and procedures are closely related to the structure of the working files, and careful thought about one can create a savings in the other. A good example of this is the format of the PLATFM array as previously shown in Figure 4. There were several ways to order the platforms. Blue and Red could have been in two groups, or the order of aircraft, surface ships and submarines could have been different. The present structure proved to be more efficient when considering the detection procedures. UP SCOPE! allows aircraft and surface ships to detect submarines, and it allows submarines to detect surface ships and other submarines. Surface ships cannot detect each other, and no platform can detect aircraft. The structure of PLATFM allows one loop to process each of the two detection phases. Other PLATFM formats would require numerous smaller loops to accomplish the same task. The structure of PLATFM also provides a counter example. The loading of the platforms is a relatively efficient section because it loads aircraft

automatically according to the basic data in Appendix B, but the fact that aircraft appear first in PLATFM required a great deal of index and pointer manipulation in the loading routine. It also resulted in a gap between aircraft and surface ships that had to be closed before the game could be started. All of these problems could have been avoided if the order of the platforms had been reversed. Having submarines first and aircraft last would have caused no change in the procedures, but would have greatly simplified the loading. Unfortunately once this was discovered too many changes in other parts of the program were required, and the effort would have been counter-productive.

The columns of PLATFM provide another double-edged example. The order of the information in each row is evidence that several additions were made after the fact. This has no effect at all on the program, and can be ignored from the player's point of view. One aspect of the row ordering that should not be ignored is the contiguous placement of certain types of information. For example, the placement of the ship's shot results in columns 35, 36 and 37 allow the use of one DO LOOP to report the results of all the weapons. Similar ordering is used for the weapon and aircraft index numbers and number of each type carried by the platform.

3. Limitations

Another aspect of data structures that should be considered by the programmer is the possible restrictions it imposes. All of the working file arrays in Up Scope are dimensioned to 100. This restricts the player from using more than 100 platforms in a game. The DETECT array imposes an even tighter restriction with its size of 100 because each platform can detect more than one ship. The choice of 100 was largely arbitrary, and can be changed with very little effort. It should also be noted that exceeding the limits of any array causes a message to be printed explaining the situation.

E. INPUT AND OUTPUT REQUIREMENTS

The amount and format of the information required and output by the program is decided largely by the degree of user-friendliness desired. The basic intent of this effort was to program a game that was very friendly. This resulted in numerous lines of output designed to give the player the ability to change his inputs and to clearly define his options when a selection was required from him. Unfortunately the overall effect is a program that is possibly too user-friendly because the amount of lines printed that contain information not directly related to the game is excessive. It actually slows the rate of play when a player becomes familiar with the game and its input requirements.

The above highlights the major tradeoff associated with input and output. A large amount of instructions and game information may be appropriate if the game will be played only a few times by each of a great number of players. However, a game that will be played many times by the same players should minimize the amount of extraneous information presented and requested.

VI. FUTURE ENHANCEMENTS AND CONCLUSION

A. FUTURE ENHANCEMENTS

1. Computer Assistance

The computer assistance provided by the computerized game could be improved by the use of two terminals. This would allow each player his own console, and eliminate the need for passwords. It would also add an air of realism by physically separating the players. The present structure of the program is such that only minor changes would be required if the proper system support were available. Another improvement would be the use of graphic displays to present and maintain the game board for each player. This would eliminate the last "chore" required of the players.

2. Rules and Procedures

Numerous recommendations could be made under this category. The only limitations are the size of the computer and the imagination of the programmer. Of the three presented here, the first two would have the effect of making the game more realistic. The third is strictly procedural.

a. Land Bases

The current version of Up Scope does not include the use of land based aircraft although the SPI version does and the aircraft are loaded in the data base. The

addition of land bases would require a special type of "platform" to be loaded in the PLATFM array. The loading of aircraft for this platform would have to be modified by allowing the player to choose the type of aircraft, unless the programmer designed several types of bases, each with different aircraft. These bases could be added to the data base as "ship classes" without the ability to move or be sunk.

b. Surface Warfare

As designed by SPI, "UP SCOPE!" is a submarine warfare game. Future efforts might include the expansion of the game to include surface warfare as well. This type of change would involve a great deal of work because an entirely new game would result.

c. Game Board

One procedural improvement to the game would be computerized control of the playing areas. A check could be programmed to ensure neither player moved his ships beyond the area dictated by the scenario. This would eliminate the need for the "gentleman's agreement" to stay within the specified area.

3. Input and Output

Although the amount and format of input and output is determined by the type of player, the suggestions below would improve the game for both the expert player familiar with the instructions and the new player.

a. Platform Loading

The procedure for the initial loading of ships requires three separate lines of information to be input. All three pieces could be entered on one line with the use of a subroutine like NONVRT. The subroutine would be required to break a string of characters into three separate "words" with each word a different format or mode. These "words" could then be assigned to the appropriate variables. This procedure could be used in other parts of the program as well.

b. Aircraft loading

Every time a ship class that carries aircraft is loaded, the program automatically loads the aircraft and asks the player to input a name for each aircraft. This procedure becomes quite lengthy when more than a few aircraft are carried. It would decrease the loading time if the players were given the option of letting the program supply names for the aircraft based on the user-supplied name of the ship. For example, a Virginia class ship named GINNY could have its two Sea Sprites named GIN1 and GIN2. This change would not be too difficult to make, using character manipulation in the aircraft loading routine.

B. CONCLUSION

Although the outline presented in Chapter V does not provide a detailed checklist including step by step instructions for converting a tactical board game, it is

entirely sufficient. More detail would not necessarily be an improvement. As it is, the outline will provide the programmer contemplating a conversion a great deal of assistance. It supplies a structure that will guide and help organize his thoughts and efforts. If the outline had been available prior to the conversion of "UP SCOPE!", it would have resulted in the improvements discussed in Chapter V, and it would have saved a great deal of time spent reprogramming routines that had to be changed because they did not "fit" the rest of the program. The outline also provides a framework for future enhancements to the program.

The military student of war gaming and the programmer required to convert a board game will find this thesis highly beneficial. It not only provides a structure that will help the programmer, it gives the student some knowledge about designing, developing and programming war games that are highlighted by actually playing "UP SCOPE!."

APPENDIX A

SPECIFIC SCENARIOS

A. SCENARIO RESTRICTIONS

Each of these three scenarios constitutes a distinct game utilizing unique forces and special rules as required. A scenario normally ends at the conclusion of the full number of Game-Turns indicated in the scenario rules; however any of these three scenarios may be prematurely concluded during the victory determination phase (at the end of the game-turn) if any of the following conditions are satisfied.

1. All of the participating surface ships have been sunk or have exited the map boundarios.

2. All the submarines have either been sunk or have exited the map boundaries.

3. Either played has achieved a major victory and the opposition desires to terminate the game.

During the three scenarios that follow, one player is designated the "Attacker" and the other the "Defender". The attacker's mission is generally to inflict the maximum possible damage on the defender's platform(s). The defender's mission is generally to ensure that his platform(s) exit from the map boundary without suffering any damage and/or to inflict the maximum possible damage on the attacking platform(s). Either players platform's are permitted to exit the map boundaries but they may not later reenter the

game. It should be noted, however, that the defender receives final Victory Points for the durability value of each defending platform remaining on the map at the conclusion of the game, if there are no attacking platforms present on the map at the end of the game.

B. SCENARIO VICTORY DETERMINATION PROCEDURE

1. At the beginning of the game utilizing one of these three scenarios, the defender records the total durability value of all platform(s) under his control and records this Initial Victory Point Total.

2. At the end of the designated number of game turns, the defender records the total durability value for all platform(s) under his control which has exited the destination map boundary as specified in the scenario utilized.

3. The total derived above is incremented by one for each durability point of damage inflicted on the opposing platform(s) during the scenario played.

4. The total derived above is the final victory point total and is then divided by the initial victory point total and expressed as a percentage. This is the victory percentage ratio. The winner and level of victory achieved are then obtained from the following:

Victory Percentage Ratio	Victor	Level of Victory
200%	defender	Major
100 to 199%	defender	Minor
50 to 99%	attacker	Minor
1 to 49%	attacker	Major
0%	attacker	Total

C. SCENARIOS

1. Here it Comes, Chicago!

a. Scenario description. On April 2, 1982, a Russian Delta class ballistic missile submarine is cruising off the northeast coast of the United States. For the last ten days the Soviet crew has been aware of, and hounded by, an American task force of two Spruance class destroyers and one Knox class escort. For the past 24 hours all platforms have been on special alert as the two superpowers approach the brink of nuclear exchange. Should hostilities commence, the Delta will attempt to obliterate a large chunk of real estate centered at Chicago!

b. Date: April 2, 1982

c. Water condition: Normal

d. Map format: 33 by 45 hexes

e. Game length: Ten Game-Turns

f. Special Rules:

(1) During the command phase of each game turn, the U.S. player rolls a die keeping the result from the Soviet

player. No platform may execute a combat mission until a six has been rolled. Once the six is rolled, all surface platforms may be assigned a combat mission according to the standard game rules. Once the U.S. player executes a combat mission, the Delta may be assigned combat missions according to the standard game rules.

(2) The Sea Sprite (Lamps) helicopters are not considered operational during this scenario.

(3) No platform may exit the board boundary during the scenario.

(4) If the Delta has not been sunk by the end of the tenth game turn, the U.S. player automatically loses both the game and Chicago.

g. Soviet/defender: One Delta class ballistic missile submarine; initial deployment anywhere within three hexes of hex 1723.

h. U.S./attacker: Two Spruances class destroyers, one Knox class escort deployed anywhere within 15 hexes of the western board boundary.

2. NATO and the Denmark Strait

a. Scenario Description: in order to interdict NATO shipping during periods of hostilities, Soviet submarines must transit several straits all of which are patrolled by NATO warships. An undeclared "hot" war commenced today, 26 June 1982.

A flotilla of Soviet submarines stationed in the Greenland Sea has been directed to move south of the Arctic circle for passage into the Atlantic. Most have orders to transit between Iceland and the Faeroe Islands. However, some attempt to make a more treacherous transit between Iceland and Greenland - through the Denmark Strait. NATO is rapidly deploying an ASW hunter-killer task group tasking them to intercept and destroy all submarines transiting the Denmark Strait. The task group consists of U.S., West German and Great Britain destroyers. This NATO task group must attempt to prevent the Soviets from making a successful passage into the Atlantic where their cruise missiles would wreak havoc on NATO shipping.

b. Date: June 28, 1982

c. Water condition: Normal

d. Map format: 33 by 90 hexes

e. Game length: 20 game turns

f. Special rules: The scenario automatically ends if at the end of any game turn either all the Soviet submarines have been sunk or exited off the western board boundary.

g. Soviet/defender: At the start of the scenario the Soviet player privately rolls a die three times, for each roll of four or less the player controls an Echo class submarine. The submarine mission is to transit the playing surface from east to west. The submarines start anywhere within five hexes of the eastern board boundary.

h. NATO/Attacker: One U.S. Spruance class destroyer, one West German Hamburg class destroyer and one Great Britain Leander class destroyer. The three are deployed anywhere within three hexes of the western board boundary.

3. Sink the America

a. Scenario description: Soviet submarines stalk U.S. aircraft carriers all over the world. In the event of hostilities, the Soviets can be expected to immediately engage our forward deployed big-deck aircraft carriers. In this scenario, the USS America is proceeding towards the Middle East war zone escorted by two modern cruisers and one escort.

b. Date: July 4, 1981

c. Water conditions: Erratic

d. Map format: 33 by 90 hexes

e. Game length: 20 game turns

f. Special rules: The aircraft aboard the USS America are not considered operational during this scenario. The helicopters aboard the cruisers and escort are operational. The scenario automatically ends during the victory determination phase if the USS America exits the eastern board boundary or if both Soviet submarines are sunk.

g. Soviet/attacker: Two Charlie class Soviet submarines deployed anywhere within ten hexes of the eastern board boundary.

h. U.S./defender: USS America, America class aircraft carrier deployed anywhere on the western board boundary, at least tex hexes from both northern and southern board boundaries. USS Virginia, Virginia class cruiser; USS South Carolina, Virginia class cruiser; USS T C Hart, Knox class escort; cruisers and escort are deployed within 5 hexes of the USS America.

APPENDIX B

SHIP DATA SUMMARY

NO: INDEX NUMBER OF SHIP CLASS
SHIP CLASS: NAME OF SHIP CLASS
SP: MAX SPEED
AD: ACTIVE DETECTION VALUE
PD: PASSIVE DETECTION VALUE
SS: ACTIVE AND PASSIVE SIMULTANEOUS SEARCHES (0-NOT POSSIBLE, 1-POSSIBLE)
SD: SPEED DEGRADATION OF PASSIVE SONOR (0-DEGRADATION, 1-NO DEGRADATION)
EV: EVASION VALUE
DE: DEFENSE VALUE
DU: DURABILITY POINTS
NO×WEAPON: NUMBER AND TYPE OF WEAPON CARRIED
NO×AIRCRAFT: NUMBER AND TYPE OF AIRCRAFT CARRIED

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
UNITED STATES											
AIRCRAFT CARRIERS											
1	NIMITZ	3	0	0	0	0	2	2	3		6×SEA KING 6×VIKING
2	AMERICA	3	3	4	0	0	1	2	3		6×SEA KING 6×VIKING
3	KITTY HAWK	3	0	0	0	0	1	2	3		6×SEA KING 6×VIKING
4	FORESTAL	3	0	0	0	0	1	2	3		6×SEA KING 6×VIKING
5	MIDWAY	3	0	0	0	0	1	2	3		6×SEA KING 6×VIKING
6	ENTERPRISE	3	0	0	0	0	2	2	3		6×SEA KING 6×VIKING
CRUISERS											
7	VIRGINIA	3	5	5	1	1	4	3	2	4×ASROC 4×MK 46	2×SEA SPRITE
8	CALIFORNIA	3	5	4	1	1	4	3	2	5×ASROC 3×MK 46	2×SEA SPRITE

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
9	TRUXTON	3	5	4	1	1	4	3	2	5×ASROC 3×MK 46	2×SEA SPRITE
10	BELKNAP	4	5	4	1	1	3	3	2	3×ASROC 4×MK 46	1×SEA SPRITE
11	BAINBRIDGE	3	4	4	0	0	3	2	2	5×ASROC 4×MK 46	1×SEA SPRITE
12	LEAHY	4	4	4	0	0	3	3	2	5×ASROC 4×MK 46	
13	ALBANY	3	4	4	0	0	3	3	2	5×ASROC 4×MK 46	
14	CHICAGO	3	4	4	0	0	3	3	2	5×ASROC 4×MK 46	1×SEA SPRITE
15	LONG BEACH	4	4	4	0	0	3	3	2	5×ASROC 4×MK 46	1×SEA SPRITE
16	CLEVELAND	3	0	3	0	0	2	1	2		
DESTROYERS											
17	AEGIS	3	5	5	1	1	4	3	1	5×ASROC	2×LAMPS MK III
18	SPRUANCE	3	5	5	1	1	4	3	1	4×MK 46 5×ASROC	1×SEA SPRITE
19	MITSCHER	3	3	4	0	0	4	2	1	5×ASROC 4×MK 46	
20	FORREST SHERMAN	3	3	4	0	0	4	2	1	5×ASROC 4×MK 46	
21	COONTZ	3	4	4	0	0	4	3	1	5×ASROC 4×MK 46	
22	CHARLES F. ADAMS	3	4	4	0	0	4	2	1	5×ASROC 4×MK 46	
23	CARPENTER	4	3	4	0	0	4	2	1	5×ASROC 4×MK 46	
24	GEARING	4	3	4	0	0	4	2	1	5×ASROC 4×MK 46	

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
FRIGATES											
25	OLIVER H. PERRY	3	5	5	1	1	4	2	1	4×MK 46	2×SEA SPRITE
26	KNOX	2	5	4	1	1	4	2	1	5×ASROC 4×MK 46	1×SEA SPRITE
27	BROOKE	2	5	4	1	1	4	2	1	5×ASROC 4×MK 46	1×SEA SPRITE
28	GARCIA	2	5	4	1	1	4	2	1	5×ASROC 4×MK 46	1×SEA SPRITE
SUBMARINES (BALLISTIC MISSILE)											
29	TRIDENT	3	0	5	0	1	5	3	2	4×MK 48	
30	LAFEYETTE	2	0	5	0	1	4	2	2	4×MK 48	
31	ETHAN ALLEN	2	0	5	0	1	4	2	2	4×MK 48	
32	G. WASHINGTON	2	0	5	0	1	4	2	2	4×MK 48	
SUBMARINES (ATTACK)											
33	LOS ANGELES	3	5	5	1	1	5	3	2	3×SUBROC 4×MK 48	
34	NARWHAL	3	4	5	1	1	5	3	2	3×SUBROC 4×MK 48	
35	STURGEON	3	4	5	1	1	5	3	2	3×SUBROC 4×MK 48	
36	PERMIT	3	4	5	1	1	4	3	2	3×SUBROC 4×MK 48	
37	TULIBEE	2	4	5	1	1	5	3	2	3×SUBROC 4×MK 48	
38	SKIPJACK	2	3	4	0	0	2	2	2	3×SUBROC 4×MK 48	
39	SKATE	2	3	4	0	0	2	2	2	3×SUBROC 4×MK 48	

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
SOVIET UNION											
AIRCRAFT CARRIERS											
40	KIEV	3	4	5	1	1	2	3	3	5×SSN-14 6×MBU-2500A 4×21"	10×HORMONE
CRUISERS											
41	MOSKVA	3	4	5	1	1	3	3	2	3×FRAS-1 6×MBU-2500A 4×21"	5×HORMONE
42	KARA	3	4	5	1	1	3	3	2	4×SSN-14 6×MBU-2500A 4×DC	1×HORMONE
43	KRESTA	3	3	4	0	0	3	3	2	4×SSN-14 6×MBU-2500A 4×DC	1×HORMONE
44	KYNDA	4	3	4	0	0	3	2	2	6×MBU-2500A	
DESTROYERS											
45	KRIVAK	4	4	5	1	1	4	2	1	4×SSN-14 6×MBU-2500A	
46	KANIN	4	3	4	0	0	4	2	1	6×MBU-2500A	
47	KOTLIN	4	3	4	0	0	4	2	1	6×MBU-2500A 4×DC	
48	KASHIN	4	3	5	1	1	4	2	1	6×MBU-2500A 6×MBU-4500A 4×DC	
49	SKORY	3	3	4	0	0	4	1	1	4×DC	
FRIGATES											
50	MIRKA	3	3	4	0	0	4	1	1	7×MBU-2500A 4×16"	
51	PETYA	3	3	4	0	0	4	2	1	7×MBU-2500A	

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
52	RIGA	3	3	4	0	D	4	1	1	6×MBU-2500A 4×DC	
SUBMARINES (BALLISTIC MISSILE)											
53	DELTA	3	4	5	0	D	4	3	2	4×21"	
54	YANKEE	2	4	5	D	0	3	3	2	4×21"	
55	GOLF	2	3	4	D	0	5	D	2	4×21"	
56	HOTEL	2	3	4	0	0	3	2	2	4×21"	
SUBMARINES (ATTACK)											
57	CHARLIE	3	4	5	0	0	4	3	2	1×SSN-7 4×21"	
58	VICTOR	4	4	5	0	0	4	3	2	4×21"	
59	ECHO	2	3	4	D	0	2	2	2	1×SSN-3 4×21"	
60	NOVEMBER	2	3	4	0	0	2	2	2	3×21"	
GREAT BRITAIN											
CRUISERS											
61	INVINCIBLE	3	3	4	D	0	3	2	2		10×SEA KING
62	BRISTOL	3	3	4	D	0	3	2	2	3×IKARA 6×LIMBO	1×WASP
DESTROYERS											
63	WEAPON	3	5	5	0	0	4	2	1	4×MK 46	2×LYNX
64	SHEFFIELD	3	4	4	0	0	4	2	1	4×MK 46	1×LYNX
65	AMAZON	3	4	4	D	0	4	2	1	4×MK 46	1×LYNX
66	LEANDER	3	3	4	D	0	4	2	1	6×LIMBO 4×MK 46	1×WASP
67	ROTHESAY	3	3	4	D	D	4	2	1	6×LIMBO	1×WASP

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
68	TRIBAL	3	3	4	0	0	4	2	1	6×LIMBO	1×WASP
69	LEOPARD	3	3	4	0	0	4	2	1	6×LIMBO	
70	SALISBURY	3	3	4	0	0	4	2	1	6×LIMBO	
71	BLACKWOOD	3	3	4	0	0	4	2	1	6×LIMBO	
72	WHITBY	3	3	4	0	0	4	2	1	6×LIMBO	
SUBMARINES (BALLISTIC MISSILE)											
73	RESOLUTION	2	4	4	0	0	4	2	2	4×TIGERFISH 4×MK 20	
SUBMARINES (ATTACK)											
74	SWIFTSURE	3	4	4	0	0	4	2	2	4×TIGERFISH 4×MK 20	
75	VALIANT	3	3	4	0	0	4	2	2	4×TIGERFISH 4×MK 20	
76	DREADNOUGHT	3	3	4	0	0	4	2	2	4×TIGERFISH 4×MK 20	
77	OBERON	2	3	3	0	0	5	2	2	4×TIGERFISH 4×MK 20	
CANADA											
DESTROYERS											
78	TRIBAL	3	4	4	1	1	4	2	1	4×MK 46	2×SEA KING
79	ANNAPOLIS	3	4	4	0	0	4	2	1	4×MK 46 4×LIMBO	1×SEA KING
80	ST. LAURENT	3	4	4	0	0	4	2	1	4×MK 46 4×LIMBO	
81	MACKENSIE	3	4	4	0	0	4	2	1	4×MK 46 5×LIMBO	
82	RESTIGOUCHE	3	4	4	0	0	4	2	1	4×MK 46 5×LIMBO	

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
SUBMARINES (ATTACK)											
83	OBERON	2	3	3	0	0	5	2	2	4×TIGERFISH 4×MK 20	
AUSTRALIA											
AIRCRAFT CARRIERS											
84	MELBOURNE	2	0	0	0	0	2	1	3		5×SEA KING
DESTROYERS											
85	PERTH	4	4	4	0	0	4	2	1	4×IKARA	
86	DARING	3	3	4	0	0	4	1	1	4×LIMBO	
FRIGATES											
87	O.H. PERRY	3	5	5	1	1	4	2	1	4×MK 46	1×SEA KING
88	RIVER	3	3	4	0	0	4	1	1	3×IKARA 4×LIMBO	
SUBMARINES (ATTACK)											
89	OBERON	2	3	3	0	0	5	2	2	4×TIGERFISH 4×MK 20	
FRANCE											
AIRCRAFT CARRIERS											
90	CLEMENCEAU	3	4	4	1	0	2	2	3		10×ALIZE 5×SUPER FRELON
CRUISERS											
91	JEANNE OF ARC	3	3	4	0	0	3	2	2		6×SUPER FRELON
92	COLBERT	3	0	0	0	0	3	1	2		

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
DESTROYERS											
93	MONTCALM	3	5	5	1	1	4	3	1	5×MK L5	1×LYNX
94	DUPERRE	3	5	5	1	1	4	3	1	5×MK L5	1×LYNX
95	TOURVILLE	3	5	5	1	1	4	3	1	5×MK L5 4×MALAFON	1×LYNX
96	ACONIT	3	5	5	1	1	4	3	1	5×MK L5 4×MALAFON 6×305MM	
97	SUFFREN	3	5	5	1	1	4	3	1	5×MK L5 4×MALAFON	
98	LA GALISSONNIERE	3	5	5	1	1	4	3	1	4×MK K2 4×MALAFON	1×LYNX
99	TARTU	3	4	4	0	0	4	2	1	4×MK K2 6×375MM	
100	KERSAINT	3	4	4	0	0	4	2	1	4×MK K2 6×375MM	
101	CASABLANCA	3	5	4	1	1	4	3	1	4×MK K2 4×MALAFON 6×375MM	
FRIGATES											
102	DESTIENNE	2	4	4	1	0	4	2	1	4×MK L5 6×375MM	
103	COMM. RIVIERE	2	3	4	0	0	4	2	1	4×MK K2 6×305MM	
104	SAVOYARD	2	4	4	0	0	4	2	1	4×MK K2 6×305MM 4×DC	
105	LE BOULONNAIS	2	3	4	0	0	4	2	1	4×MK K2 6×375MM	

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
SUBMARINES (BALLISTIC MISSILE)											
106	REDOUBTABLE	2	4	4	1	0	4	3	2	5×MK E12	
SUBMARINES (ATTACK)											
107	AGOSTA	2	4	5	1	0	5	3	2	5×MK E12	
108	DAPHNE	2	4	5	1	0	5	3	2	5×MK E12	
109	ARETHUSE	2	0	4	0	0	5	0	2	5×MK E12	
110	NARVAL	2	0	4	0	0	5	0	2	5×MK E12	
ITALY											
CRUISERS											
111	VITTORIO VENETO	3	3	4	0	0	3	2	2	5×MK 44	9×AB-204
112	ANDREA DORIA	3	3	4	0	0	3	2	2	5×MK 44	4×AB-204
DESTROYERS											
113	AUDACE	4	4	4	0	0	4	3	1	5×MK 44	1×SEA KING
114	IMPAVIDO	4	3	4	0	0	4	2	1	5×MK 44	1×AB-204
115	IMPETUOSO	4	2	4	0	0	4	1	1	5×MK 44 6×MENON	
FRIGATES											
116	LUPO	3	4	4	0	0	4	2	1	5×MK 44	
117	ALPINO	2	3	4	0	1	4	2	1	5×MK 44 5×MENON	2×AB-204
118	DE CHRISTOFARO	2	3	4	0	1	4	2	1	5×MK 44 6×MENON	
119	BERGAMINI	2	3	4	0	0	4	2	1	5×MK 44	
120	CENTAURO	3	3	4	0	0	4	2	1	5×MK 44 6×MENON	

NO	SHIP CLASS	SP	AD	PD	SS	SD	EV	DE	DU	NO×WEAPON	NO×AIRCRAFT
121	ALBATROS	2	0	0	0	0	4	0	1	5×MK 44 6×HEDGEHOG	
SUBMARINES (ATTACK)											
122	SAURO	2	3	4	0	0	5	2	2	5×533MM	
123	EX-TANG	2	3	4	0	0	5	1	2	4×533MM	
124	TOTI	2	3	4	0	0	5	1	2	5×533MM	
WEST GERMANY											
DESTROYERS											
125	LUTJENS	4	4	4	0	0	4	2	1	5×MK 44 5×ASROC	
126	HAMBURG	4	3	4	0	0	4	2	1	4×533MM 6×BOFORS 375MM	
FRIGATES											
127	KOLN	3	3	4	0	0	4	2	1	4×533MM 6×BOFORS 375MM	
128	THETIS	3	3	4	0	0	4	1	1	4×533MM 6×BOFORS 375MM	
SUBMARINES (ATTACK)											
129	U-206	2	0	4	0	0	5	1	2	5×MK 37C	
130	U-205	2	0	4	0	0	5	1	2	5×MK 37C	

APPENDIX C

WEAPON DATA SUMMARY

NO: INDEX NUMBER OF THE WEAPON
NAME: NAME OF THE WEAPON
ACCURACY: ACCURACY RATING
RANGE: RANGE IN HEXES
TYPE: 1-MISSILE, 2-MISSILE/DEPTH BOMB, 3-TORPEDO, 4-DEPTH CHARGE

NO	NAME	ACCURACY	RANGE	TYPE
1	SSN-3 (USSR)	5	30	1
2	SSN-7 (USSR)	5	30	1
3	SUBROC (US)	6	25	2
4	SSN-14 (USSR)	6	20	3
5	21" (USSR)	5	15	3
6	16" (USSR)	5	10	3
7	MK 48 (US)	6	25	3
8	MK 37C (US, WG)	5	15	3
9	TIGERFISH (GB)	6	15	3
10	MK 20 (GB)	5	7	3
11	MK K2 (FR)	6	10	3
12	MK L5 (FR)	5	8	3
13	MK E12 (FR)	5	8	3
14	533MM (IT)	5	12	3
15	ASROC (US, GB)	6	7	3
16	MK 46 (US)	5	20	3
17	MK 44 (US, GB, IT)	4	12	3
18	IKARA (GB, AUS)	6	10	3
19	MALAFON (FR)	5	8	3

NO	NAME	ACCURACY	RANGE	TYPE
20	FRAS-1 (USSR)	6	16	2
21	MBU-2500A (USSR)	4	3	4
22	MBU-4500A (USSR)	4	2	4
23	LIMBO (GB)	3	1	4
24	HEDGEHOG (GB)	3	1	4
25	375MM (FR)	4	1	4
26	305MM (FR)	3	1	4
27	MENON (IT)	3	1	4
28	BOFORS 375MM (GB)	3	1	4
29	DC (ALL NATIONS)	3	1	4

APPENDIX D

AIRCRAFT DATA SUMMARY

NO: INDEX NUMBER OF THE AIRCRAFT TYPE
NAME: NAME OF THE AIRCRAFT TYPE
SP: MAX SPEED
AD: ACTIVE DETECTION VALUE
PD: PASSIVE DETECTION VALUE
NO*WEAPON: NUMBER AND TYPE OF WEAPON CARRIED

NO	NAME	SP	AD	PV	NO*WEAPON
HELICOPTERS					
1	HORMONE (USSR)	11	4	4	1*16" 3*DC
2	SEA KING (NATO)	11	4	4	1*MK 46 1*DC
3	LYNK (NATO)	15	4	4	1*MK 46
4	WASP (NATO)	10	3	3	1*MK 44
5	SEA SPRITE (US)	12	4	4	1*MK 46
6	LAMPS MK III (US)	16	4	4	1*MK 46
7	WESSEX (GB)	12	3	3	1*MK 46
8	SUPER FRELON (FR)	14	4	4	1*MK 44
9	AB-204 (IT)	10	3	3	1*MK 44
FIXED WING					
10	MAIL (USSR)	24	3	3	4*DC
11	MAY (USSR)	31	3	3	5*DC
12	TRACKER (US)	15	4	4	1*MK 46 3*DC

NO	NAME	SP	AD	PV	NO×WEAPON
13	ORION (US)	30	4	4	3×MK 46 4×DC
14	VIKING (US)	35	4	4	1×MK 46 3×DC
15	NIMROD (US)	35	4	4	3×MK 46 4×DC
16	ALIZE (FR)	24	3	3	3×MK 44 4×DC

APPENDIX E

PRACTICE GAME

```
> $ scope
EXECUTION BEGINS...

THE FOLLOWING OPTIONS ARE AVAILABLE TO YOU:


1.  ZERO ANY PREVIOUSLY ESTABLISHED DATA BASE.

2.  EXIT THE GAME.

3.  INITIALIZE GAME PARAMETERS AND COMMENCE PLAY.


ENTER AN OPTION.
> 3

YOU ARE NOW GOING TO ENTER THE BASIC GAME PARAMETERS.
ONE WORD OF CAUTION.  THE PRESENT GAME CAN ONLY HANDLE A
TOTAL OF 100 WEAPON SYSTEMS.  INCLUDED IN THESE SYSTEMS
ARE SURFACE SHIPS, SUBMARINES AND AIRCRAFT.  THE NUMBER OF
AIRCRAFT ARE DETERMINED BY THE SHIP CLASS AS REFLECTED IN THE
DATA BASE, AND WILL BE LOADED AUTOMATICALLY WHEN YOU SELECT
THE SHIP CLASS.  PRIOR TO BEGINNING THE LOADING PROCEDURE
YOU SHOULD ENSURE THAT THE TOTAL NUMBER OF EACH PLAYERS
WEAPON SYSTEMS DOES NOT EXCEED   50.


ENTER THE NUMBER OF GAME TURNS YOU WANT TO PLAY.
> 2


ENTER THE WATER CONDITION (0-NORMAL, 1-ERRATIC).
> 0

THE FOLLOWING INPUTS APPLY ONLY TO THE BLUE PLAYER
THE FIRST IS THE BLUE PLAYERS PASSWORD WHICH IS
A MAXIMUM OF 6 CHARACTERS.  THE PASSWORD IS IMPORTANT BECAUSE
IT IS REQUIRED PRIOR TO THE INPUT OF COMBAT, MOVE AND SEARCH
COMMANDS.  IF YOU FORGET THE PASSWORD, YOU WILL HAVE TO RETURN
TO MENU SELECTION, AND RESTART THE GAME.


SELECT AND ENTER YOUR PASSWORD
> blupas
```

```
YOUR PASSWORD WILL BE -BLUPAS- .  IF YOU ARE
READY TO CONTINUE TYPE Y(ES).  TYPE N(O) IF YOU WISH TO
CHANGE YOUR PASSWORD.
>y

ENTER THE NUMBER OF BLUE SURFACE SHIPS.
>2


ENTER THE NUMBER OF BLUE SUBMARINES.
>1


ENTER THE NUMBER OF BLUE SHIPS THAT MUST
BE DESTROYED FOR PREMATURE GAME ENDING
>2


ECHO

BLUE SURFACE SHIPS -     2
BLUE SUBMARINES -    1
NUMBER OF BLUE SHIPS DESTROYED TO END GAME -      2



TYPE Y(ES) IF THESE VALUES ARE CORRECT, OR N(O) IF
YOU WANT TO REENTER THEM.
>y

THE FOLLOWING INPUTS APPLY ONLY TO THE RED PLAYER
THE FIRST IS THE RED PLAYERS PASSWORD WHICH IS A
MAXIMUM OF 6 CHARACTERS.  THE PASSWORD IS IMPORTANT BECAUSE
IT IS REQUIRED PRIOR TO THE INPUT OF COMBAT, MOVE AND SEARCH
COMMANDS.  IF YOU FORGET THE PASSWORD, YOU WILL HAVE TO RETURN
TO MENU SELECTION, AND RESTART THE GAME.


SELECT AND ENTER YOUR PASSWORD.
>redpas


YOUR PASSWORD WILL BE -REDPAS- .  IF YOU ARE
READY TO CONTINUE TYPE Y(ES).  TYPE N(O) IF YOU WISH TO
CHANGE YOUR PASSWORD.
>y.


ENTER THE NAME OF RED SURFACE SHIPS.
>0
```

```
ENTER THE NUMBER OF RED SUBMARINES.
>2


ENTER THE NUMBER OF RED SHIPS THAT MUST
BE DESTROYED FOR PREMATURE GAME ENDING.
>2


ECHO

RED SURFACE SHIPS -    0
RED SUBMARINES -    2
NUMBER OF RED SHIPS DESTROYED TO END GAME -     2


TYPE Y(ES) IF THESE VALUES ARE CORRECT, OR N(O) IF
YOU WANT TO REENTER THEM.
>y

YOU ARE NOW READY TO LOAD THE BLUE WEAPON SYSTEMS.

ENTER THE BLUE PASSWORD.
>blupas

BLUE SURFACE SHIP NUMBER     1

ENTER A 6 CHARACTER NAME FOR THE SHIP.
>tincan

ENTER THE SHIP CLASS OF TINCAN
>spruance

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF TINCAN
>500


THE SPRUANCE      CLASS OF SHIP HAS 1 SEA SPRITE TYPE AIRCRAFT.

ENTER A 6 CHARACTER NAME FOR AIRCRAFT   1
>sprul

BLUE SURFACE SHIP NUMBER 2

ENTER A 6 CHARACTER NAME FOR THE SHIP.
>escort

ENTER THE SHIP CLASS OF ESCORT
>knox
```

```
ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF ESCORT
>520


THE KNOX    CLASS OF SHIP HAS 1 SEA SPRITE  TYPE AIRCRAFT.

ENTER A 6 CHARACTER NAME FOR AIRCRAFT    1
>knox1

BLUE SUBMARINE NUMBER     1

ENTER A 6 DIGIT CHARACTER NAME FOR THE SUBMARINE.
>lapon

ENTER THE SUBMARINE CLASS OF LAPON
>sturgeon

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF LAPON
>510

YOU ARE NOW READY TO LOAD THE RED WEAPON SYSTEMS.

ENTER THE RED PASSWORD.
>redpas

RED SUBMARINE NUMBER   1

ENTER A 6 DIGIT CHARACTER NAME FOR THE SUBMARINE.
>ivan

ENTER THE SUBMARINE CLASS OF IVAN
>charlie

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF IVAN
>1510

RED SUBMARINE NUMBER    2

ENTER A 6 DIGIT CHARACTER NAME FOR THE SUBMARINE.
>boris

ENTER THE SUBMARINE CLASS OF BORIS
>november

ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL
LOCATION OF BORIS
>1520
```

```
BLUE CALL TO STATUS ROUTINES.  ENTER THE BLUE PASSWORD.
>blupas

THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:


1.  DETECTIONS OF ENEMY PLATFORMS.

2.  RESULTS OF WEAPONS FIRED.

3.  DAMAGE TO FRIENDLY PLATFORMS.

4.  STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.

5.  BEGIN NEXT GAME TURN.


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4

YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT,MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.

>tincan


TINCAN IS LOCATED AT POSITION  500 SPEED 0
IT WAS NOT ASSIGNED A SEARCH MISSION
TINCAN WEAPON SYSTEM(S)
   TYPE      RANGE
1  MK 46       20
2  ASROC        7


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION       IN THAT ORDER, OTHERWISE ENTER N(ONE)

>2
>502
```

```
ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>3

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
         NONE SCHEDULED
DESTINATION 502   SPEED 2
ACTIVE AND PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>escort


ESCORT IS LOCATED AT POSITION  520 SPEED 0
IT WAS NOT ASSIGNED A SEARCH MISSION
ESCORT WEAPON SYSTEM(S)
    TYPE     RANGE
1   ASROC      7
2   MK 46     20


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n

TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION   IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>720
```

```
ENTER SEARCH MISSIONS AS FOLLOWS:
    0  NONE SCHEDULED
    1  ACTIVE MISSION
    2  PASSIVE MISSION
    3  ACTIVE AND PASSIVE


>1

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
       NONE SCHEDULED
DESTINATION  720     SPEED 2
ACTIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>lapon


LAPON IS LOCATED AT POSITION  510 SPEED 0
IT WAS NOT ASSIGNED A SEARCH MISSION
LAPON WEAPON SYSTEM(S)
   TYPE    RANGE
1  SUBROC    25
2  MK 48     25


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION      IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>512
```

```
ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


 >2

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        NONE SCHEDULED
DESTINATION 512    SPEED 2
PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
 >y


ENTER THE NAME OF THE DESIRED PLATFORM


 >knox1


KNOX1 IS CURRENTLY NOT FLYING - IT IS ASSUMED YOU
DESIRED TO LAUNCH.


KNOX1 IS NOW BEING LAUNCHED


MOVEMENT NOT DESIRED ENTER N(ONE)  OTHERWISE ENTER
SPEED
DESTINATION HEX      IN THIS FASHION


 >12
 >1720


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION


 >0
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
```

```
NONE SCHEDULED
DESTINATION 1720 SPEED 12
THERE IS NO SEARCH MISSION SCHEDULED


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>n


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>5

THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)
IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH
INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)
TO RETURN TO DISPLAY SELECTION.
>y

RED CALL TO STATUS ROUTINES.  ENTER THE RED PASSWORD.
>redpas

THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:


1.  DETECTIONS OF ENEMY PLATFORMS.

2.  RESULTS OF WEAPONS FIRED.

3.  DAMAGE TO FRIENDLY PLATFORMS.

4.  STATUSOF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.

5.  BEGIN NEXT GAME TURN.


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4

YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT, MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.


>ivan
```

```
IVAN   IS LOCATED AT POSITION 1510 SPEED 0
IT WAS NOT ASSIGNED A SEARCH MISSION
IVAN   WEAPON  SYSTEM(S)
    TYPE    RANGE
1   SSN-7     30
2   21"       15


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION     IN THAT ORDER, OTHERWISE ENTER N(ONE)

>1
>1511


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>2

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
      NONE SCHEDULED
DESTINATION 1511   SPEED 1
PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>boris
```

```
BORIS IS LOCATED AT POSITION 1520 SPEED 0
IT WAS NOT ASSIGNED A SEARCH MISSION
BORIS  WEAPON SYSTEM(S)
    TYPE     RANGE
1   21"        15


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION     IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>1522


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>1

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
       NONE SCHEDULED
DESTINATION 1522    SPEED 2
ACTIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>n

SELECT AND ENTER THE NUMBER FOR A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>5
```

```
THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)
IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH
INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)
TO RETURN TO DISPLAY SELECTION.
>y

BLUE CALL TO STATUS ROUTINES.  ENTER THE BLUE PASSWORD.
>blupas

THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:


1.  DETECTIONS OF ENEMY PLATFORMS.

2.  RESULTS OF WEAPONS FIRED.

3.  DAMAGE TO FRIENDLY PLATFORMS.

4.  STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.

5.  BEGIN NEXT GAME TURN.


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>1

             SONOR          SHIP
DETECTING    TYPE           TYPE
PLATFORM     USED        DETECTED     LOCATION

  TINCAN     ACTIVE      SUBMARINE       512
  ESCORT     ACTIVE      SUBMARINE       512
  ESCORT     ACTIVE      SUBMARINE      1522
  TINCAN     PASSIVE     SUBMARINE      1522
  LAPON      PASSIVE     SURFACE         502
  LAPON      PASSIVE     SURFACE         720
  LAPON      PASSIVE     SUBMARINE      1511
  LAPON      PASSIVE     SUBMARINE      1522

END OF DETECTION ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y

SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4
```

```
YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT,MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.


>tincan


TINCAN IS LOCATED AT POSITION  502 SPEED 2
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE AND PASSIVE
    TYPE     LOCATION      METHOD      RANGE
SUBMARINE      512         ACTIVE       10
SUBMARINE     1522         PASSIVE      25
TINCAN WEAPON SYSTEM(S)
    TYPE     RANGE
1   MK 46     20
2   ASROC      7


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION       IN THAT ORDER, OTHERWISE ENTER N(ONE)


>3
>802


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>3

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        NONE SCHEDULED
DESTINATION  802    SPEED 3
ACTIVE AND PASSIVE SEARCH MISSION
```

```
ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y

ENTER THE NAME OF THE DESIRED PLATFORM

>escort


ESCORT IS LOCATED AT POSITION  720 SPEED 2
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
    TYPE      LOCATION       METHOD       RANGE
SUBMARINE        512         ACTIVE         9
SUBMARINE       1522         ACTIVE         8
ESCORT WEAPON SYSTEM(S)
    TYPE      RANGE
1  ASROC         7
2  MK 46        20


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>1522
>1
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)

>2

TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)

>n

TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION    IN THAT ORDER, OTHERWISE ENTER N(ONE)
>2
>72

YOU HAVE EITHER SET A SPEED HIGHER THAN MAX CAPABLE OR
ATTEMPTED TO GO FURTHER THAN THE SPEED SETTING WILL ALLOW

TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION     IN THAT ORDER, OTHERWISE ENTER N(ONE)

>2
>1722
```

```
YOU HAVE EITHER SET A SPEED HIGHER THAN MAX CAPABLE OR
ATTEMPTED TO GO FURTHER THAN THE SPEED SETTING WILL ALLOW


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION     IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>722


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>3

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        ASROC        WEAPON    TGT LOCATION 1522
        MK 46        WEAPON    TGT LOCATION 1522
DESTINATION  722     SPEED 2
ACTIVE AND PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>lapon


LAPON IS LOCATED AT POSITION 512 SPEED 2
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
PASSIVE
   TYPE        LOCATION      METHOD        RANGE
SURFACE          502         PASSIVE         10
SURFACE          720         PASSIVE          9
SUBMARINE       1511         PASSIVE         10
SUBMARINE       1522         PASSIVE         15
```

```
LAPON WEAPON SYSTEM(S)
    TYPE      RANGE
1   SUBROC      25
2   MK 48       25


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


 >1522
 >1
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


 > 2
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


 >n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION       IN THAT ORDER, OTHERWISE ENTER N(ONE)


 >2
 >712


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


 >1


YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        SUBROC       WEAPON    TGT LOCATION 1522
        MK 48        WEAPON    TGT LOCATION 1522
DESTINATION  712     SPEED 2
ACTIVE SEARCH MISSION
```

```
ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y

ENTER THE NAME OF THE DESIRED PLATFORM

>spru1

SPRU1  IS CURRENTLY NOT FLYING - IT IS ASSUMED YOU
DESIRE TO LAUNCH.

SPRU1  IS NOW BEING LAUNCHED

MOVEMENT NOT DESIRED ENTER N(ONE)  OTHERWISE ENTER
SPEED
DESTINATION HEX    IN THIS FASHION

>10
>1502

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION

>0
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
NONE SCHEDULED
DESTINATION 1502 SPEED 10
THERE IS NO SEARCH MISSION SCHEDULED

ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y

ENTER THE NAME OF THE DESIRED PLATFORM

>knox1
```

```
KNOX1   IS LOCATED AT POSITION 1720 AT SPEED 12
IT WAS NOT ASSIGNED A SEARCH MISSION


NO COMBAT ENTER N(ONE)   OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n
MOVEMENT NOT DESIRED ENTER N(ONE)  OTHERWISE ENTER
SPEED
DESTINATION HEX     IN THIS FASHION


>n

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION


>2
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
NONE SCHEDULED
SPEED IS ZERO AT LOCATION 1720
A PASSIVE SEARCH MISSION WILL BE CONDUCTED


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>n


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>5


THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)
IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH
INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)
TO RETURN TO DISPLAY SELECTION.
>y


RED CALL TO STATUS ROUTINES.  ENTER THE RED PASSWORD.
>redpas
```

```
THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:

1.  DETECTIONS OF ENEMY PLATFORMS.

2.  RESULTS OF WEAPONS FIRED.

3.  DAMAGE TO FRIENDLY PLATFORMS.

4.  STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.

5.  BEGIN NEXT GAME TURN.

SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>1

                SONOR        SHIP
DETECTING       TYPE         TYPE
PLATFORM        USED       DETECTED      LOCATION

   IVAN        PASSIVE     SURFACE          502
   IVAN        PASSIVE     SURFACE          720
   IVAN        PASSIVE     SUBMARINE       1522

END OF DETECTION ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4


YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT,MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.


>ivan


IVAN   IS LOCATED AT POSITION 1511 SPEED 1
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
PASSIVE
    TYPE        LOCATION      METHOD        RANGE
SURFACE           502        PASSIVE          14
SURFACE           720        PASSIVE          13
SUBMARINE        1522        PASSIVE          11
```

```
IVAN    WEAPON SYSTEM(S)
   TYPE        RANGE
1  SSN-7         30
2  21"           15


NO COMBAT ENTER N(ONE) - OTHERWISE
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>502
>1
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


>2
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
 OTHERWISE ENTER N(O)


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION    IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>1311


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>1


YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        SSN-7       WEAPON  TGT LOCATION 502
        21"         WEAPON  TGT LOCATION 502
DESTINATION 1311    SPEED 2
ACTIVE SEARCH MISSION
```

```
ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
> y


ENTER THE NAME OF THE DESIRED PLATFORM


>boris



BORIS IS LOCATED AT POSITION 1522 SPEED 2
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
    TYPE      LOCATION     METHOD        RANGE
THERE WERE NO DETECTIONS
BORIS WEAPON SYSTEM(S)
    TYPE       RANGE
1   21"        15


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION      IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>1722


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>1


YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
        NONE SCHEDULED
```

```
DESTINATION 1722    SPEED 2
ACTIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>n


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS
>y

YOU HAVE MADE AN ERROR IN YOUR INPUT SELECTION.  TRY AGAIN.


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>5

THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)
IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH
INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)
TO RETURN TO DISPLAY SELECTION.
>y
THE ATTACK BY ESCORT FAILED AS TGT RANGE> WEAPON RANGE

YOU HAVE JUST COMPLETED GAME TURN   2.
TYPE Y(ES) IF YOU WANT TO CONTINUE THE GAME, OR N(O)
TO ALLOW THE GAME TO END.
>y

BLUE CALL TO STATUS ROUTINES.  ENTER THE BLUE PASSWORD.
>blupas

THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:


1.  DETECTIONS OF ENEMY PLATFORMS.

2.  RESULTS OF WEAPONS FIRED.

3.  DAMAGE TO FRIENDLY PLATFORMS.

4.  STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.

5.  BEGIN NEXT GAME TURN.
```

```
SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>3

      SHIP        STATUS/HITS

     TINCAN              0

     ESCORT              0

     LAPON               0

END OF DAMAGE DISPLAY ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>2


TYPE Y(ES) FOR SHIP RESULTS.
>y


SHIP - ESCORT    COMBAT LOCATION - 1522
WEAPON 1   TYPE - ASROC  PROB(HIT)/DETONATION - NO DETONATION
WEAPON 2   TYPE - MK 46  PROB(HIT)/DETONATION - DETONATION


SHIP - LAPON     COMBAT LOCATION - 1522
WEAPON 1   TYPE - SUBROC PROB(HIT)/DETONATION - 0.6667
WEAPON 2   TYPE - MK 48  PROB(HIT)/DETONATION - DETONATION

END OF WEAPON RESULT DISPLAY.  TYPE Y(ES) TO CONTINUE.
>y


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>1

           SONOR         SHIP
DETECTING   TYPE         TYPE
PLATFORM    USED      DETECTED       LOCATION

ESCORT     ACTIVE    SUBMARINE          712
KNOX1      PASSIVE   SUBMARINE          712
KNOX1      PASSIVE   SUBMARINE         1311
```

```
TINCAN      PASSIVE     SUBMARINE       712
TINCAN      PASSIVE     SUBMARINE      1311
ESCORT      PASSIVE     SUBMARINE       712
ESCORT      PASSIVE     SUBMARINE      1311
LAPON       ACTIVE      SUBMARINE      1311

END OF DETECTION ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4


YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT,MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.


>tincan


TINCAN IS LOCATED AT POSITION  802 SPEED 3
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE AND PASSIVE
   TYPE      LOCATION       METHOD      RANGE
SUBMARINE      712         PASSIVE       10
SUBMARINE     1311         PASSIVE       11
TINCAN WEAPON SYSTEM(S)
   TYPE     RANGE
1  MK 46     20
2  ASROC      7


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>1311
>1
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION  IN THAT ORDER, OTHERWISE ENTER N(ONE)
```

```
> 3
> 1102


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>3

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
       MK 46     WEAPON  TGT LOCATION 1311
DESTINATION 1102 SPEED 3
ACTIVE AND PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>escort



ESCORT IS LOCATED AT POSITION  722 SPEED 2

    THE ESCORT ATTACK ON TARGET LOCATED AT 1522

RELEASE OF ASROC    RESULTED IN
NO DETONATION - NO DISTURBED WATER MARKER

RELEASE OF MK 46      RESULTED IN
A DETONATION(HIT)


IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE AND PASSIVE
    TYPE       LOCATION      METHOD       RANGE
SUBMARINE        712         ACTIVE        10
SUBMARINE        712         PASSIVE       10
SUBMARINE       1311         PASSIVE       14
ESCORT WEAPON SYSTEM(S)
    TYPE       RANGE
1   ASROC        7
2   MK 46       20
```

```
NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER   IN THIS FASHION


 >1311
 >2
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION     IN THAT ORDER, OTHERWISE ENTER N(ONE)


 >2
 >720


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


 >3

YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
      MK 46    WEAPON   TGT LOCATION 1311
DESTINATION   720   SPEED 2
ACTIVE AND PASSIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
 >y


ENTER THE NAME OF THE DESIRED PLATFORM


 >lapon


LAPON IS LOCATED AT POSITION  712 SPEED 2
```

```
    THE LAPON ATTACK ON TARGET LOCATED AT 1522

RELEASE OF SUBROC     RESULTED IN
A DETONATION WITH PROBABILITY
OF A HIT OF 4 OUT OF 6


RELEASE OF MK 48    RESULTED IN
A DETONATION(HIT)

IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
    TYPE      LOCATION      METHOD       RANGE
SUBMARINE       1311        ACTIVE         6
LAPON WEAPON SYSTEM(S)
    TYPE        RANGE
1  SUBROC        25
2  MK 48         25


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>1311
>2
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


>n

TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION    IN THAT ORDER, OTHERWISE ENTER N(ONE)

>2
>214

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE

>1
```

```
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
    MK 48    WEAPON TGT LOCATION 1311
DESTINATION 714   SPEED 2
ACTIVE SEARCH MISSION


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>spru1


SPRU1 IS LOCATED AT POSITION 1502 AT SPEED 10
IT WAS NOT ASSIGNED A SEARCH MISSION


NO COMBAT ENTER N(ONE)    OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>n
MOVEMENT NOT DESIRED ENTER N(ONE)    OTHERWISE ENTER
SPEED
DESTINATION HEX      IN THIS FASHION


>n

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION


>2


YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
NONE SCHEDULED
SPEED IS ZERO AT LOCATION 1502
A PASSIVE SEARCH MISSION WILL BE CONDUCTED
```

```
ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y


ENTER THE NAME OF THE DESIRED PLATFORM


>knox11
YOU HAVE ENTERED AN INCORRECT PLATFORM NAME.
   REENTER



>knox1


KNOX1 IS LOCATED AT POSITION 1720 AT SPEED   0
IT CONDUCTED A PASSIVE SEARCH WITH FOLLOWING RESULTS
SUBMARINE CONTACT LOCATION      RANGE
         712                     13
        1311                     11
KNOX1 WEAPON SYSTEM(S)
   TYPE       RANGE
1 MK 46        20


NO COMBAT ENTER N(ONE)    OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER    IN THIS FASHION


>1311
>1
MOVEMENT NOT DESIRED ENTER N(ONE)  OTHERWISE ENTER
SPEED
DESTINATION HEX    IN THIS FASHION


>n

ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION


>1
YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
MK 46     WEAPON     TGT LOCATION  1311
```

```
SPEED IS ZERO AT LOCATION 1720
AN ACTIVE SEARCH MISSION WILL BE CONDUCTED


ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
> n


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
> 5

THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)
IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH
INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)
TO RETURN TO DISPLAY SELECTION.
>y

RED CALL TO STATUS ROUTINES.  ENTER THE RED PASSWORD.
> redpas

THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:

1.  DETECTIONS OF ENEMY PLATFORMS.

2.  RESULTS OF WEAPONS FIRED.

3.  DAMAGE TO FRIENDLY PLATFORMS.

4.  STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.

5.  BEGIN NEXT GAME TURN.


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>3

       SHIP          STATUS/HITS

       IVAN                0

       BORIS           SUNK

END OF DAMAGE DISPLAY ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y
```

```
SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>2


SHIP - IVAN      COMBAT LOCATION - 502
WEAPON 1   TYPE - SSN-7  PROB(HIT)/DETONATION - 0.1667
WEAPON 2   TYPE - 21"    PROB(HIT)/DETONATION-NO DETONATION

END OF WEAPON RESULT DISPLAY.  TYPE Y(ES) TO CONTINUE.
>y


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>1

                 SONOR          SHIP
DETECTING        TYPE           TYPE
PLATFORM         USED         DETECTED      LOCATION

   IVAN          ACTIVE       SUBMARINE       712

END OF DETECTION ROUTINE.  TYPE Y(ES) TO CONTINUE.
>y


SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.
>4


YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY AND
ASSOCIATED COMBAT,MOVEMENT AND SEARCH COMMAND INPUTS.
ENTER THE NAME OF THE DESIRED PLATFORM.


 >ivan


IVAN    IS LOCATED AT POSITION 1311 SPEED 2

    THE IVAN    ATTACK ON TARGET LOCATED AT   502

RELEASE OF SSN-7   RESULTED IN
A DETONATION WITH PROBABILITY
OF A HIT OF 1 OUT OF 6
```

```
RELEASE OF 21"        RESULTED IN
NO DETONATION - NO DISTURBED WATER MARKER

IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
    TYPE        LOCATION         METHOD         RANGE
SUBMARINE          712           ACTIVE          6
IVAN      WEAPON SYSTEM(S)
    TYPE          RANGE
1  SSN-7            30
2  21"              15


NO COMBAT ENTER N(ONE) - OTHERWISE ENTER
TARGET LOCATION
WEAPON SYSTEM NUMBER     IN THIS FASHION


>712
>2
TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER
OTHERWISE ENTER N(O)


>n


TO SCHEDULE MOVEMENT ENTER
SPEED
DESTINATION       IN THAT ORDER, OTHERWISE ENTER N(ONE)


>2
>1111


ENTER SEARCH MISSIONS AS FOLLOWS:
     0  NONE SCHEDULED
     1  ACTIVE MISSION
     2  PASSIVE MISSION
     3  ACTIVE AND PASSIVE


>1


YOU HAVE ENTERED THE FOLLOWING COMMANDS
COMBAT:
       21"        WEAPON  TGT LOCATION    712
DESIGNATION 1111    SPEED 2
ACTIVE SEARCH MISSION
```

```
ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>y

ENTER THE NAME OF THE DESIRED PLATFORM

>boris

BORIS IS LOCATED AT POSITION 1522 SPEED 2

        NOTE
DUE TO PREVIOUS AND/OR CURRENT DAMAGE SUFFERED, MAX SPEED
IS NOW 0 THE SHIP HAS BEEN HIT 3 TIMES

IT HAS BEEN HIT BY A TORPEDO
IT HAS BEEN HIT BY A NON-TORPEDO WEAPON
IT HAS BEEN HIT BY A TORPEDO
IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S)
ACTIVE
    TYPE     LOCATION      METHOD        RANGE
THERE WERE NO DETECTIONS

SINCE THE BORIS HAS BEEN SUNK, NO COMMAND INPUTS CAN BE
ACCEPTED

ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM
ENTER N(O) TO RETURN TO DISPLAY SELECTION
>n

SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.
ENTER A 6 FOR A LIST OF DISPLAY OPTIONS
>5

THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)
IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH
INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)
TO RETURN TO DISPLAY SELECTION.
>y
```

```
BLUE RESULTS:

      SHIP          STATUS/HITS          FINAL LOCATION

     TINCAN              0                  1102

     ESCORT              0                   720

     LAPON               0                   714

TYPE Y(ES) TO SEE RED RESULTS.
>y

RED RESULTS:

     SHIP          STATUS/HITS          FINAL LOCATION

     IVAN             SUNK                 1311

     BORIS            SUNK                 1522

TYPE Y(ES) TO RETURN TO MENU SELECTION AND START A NEW
GAME.  TYPE N(O) TO EXIT THE GAME.
>y

THE FOLLOWING OPTIONS ARE AVAILABLE TO YOU:


1.  ZERO ANY PREVIOUSLY ESTABLISHED DATA BASE.

2.  EXIT THE GAME.

3.  INITIALIZE GAME PARAMETERS AND COMMENCE PLAY.


ENTER AN OPTION.
>2

EXISTING THE GAME.
R;
```

APPENDIX F
PROGRAM LISTING

```
1            IMPLICIT INTEGER (A-Z)
2            DIMENSION SHIPOA(130,18), NAMESH(130,8), ACDATA(16,7), NAMEAC(16,
3           16),WEPDA(29,3),NAMEWP(29,6),S1(130),S2(130),S3(130),S4(130),
4           1S5(130),S6(130),S7(130),S8(130),S9(130),S10(130),S11(130),
5           1S12(130),S13(130),S14(130),S15(130),S16(130),S17(190),
6           1S18(130),NS1(130),NS2(130),NS3(130),NS4(130),NS5(130),
7           1NS6(130),NS7(130),NS8(130),
8           1INPUT(70),ALPHA(2),CHECK(6),CLASNA(8)
9            EQUIVALENCE (SHIPOA(1,1),S1(1)),(SHIPOA(1,2),S2(1)),
10          1(SHIPOA(1,3),S3(1)),(SHIPOA(1,4),S4(1)),(SHIPOA(1,5),S5(1)),
11          1(SHIPOA(1,6),S6(1)),(SHIPOA(1,7),S7(1)),(SHIPOA(1,8),S8(1)),
12          1(SHIPOA(1,9),S9(1)),(SHIPOA(1,10),S10(1)),(SHIPOA(1,11),S11(1)),
13          1(SHIPOA(1,12),S12(1)),(SHIPOA(1,13),S13(1)),(SHIPOA(1,14),S14(1)),
14          1(SHIPOA(1,15),S15(1)),(SHIPOA(1,16),S16(1)),(SHIPOA(1,17),S17(1)),
15          1(SHIPOA(1,18),S18(1)),(NAMESH(1,1),NS1(1)),(NAMESH(1,2),NS2(1)),
16          1(NAMESH(1,3),NS3(1)),(NAMESH(1,4),NS4(1)),(NAMESH(1,5),NS5(1)),
17          1(NAMESH(1,6),NS6(1)),(NAMESH(1,7),NS7(1)),(NAMESH(1,8),NS8(1))
18    C
19    C     COMMON STATEMENT
20    C
21           COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
22          1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
23          2IPLAT,IAIR,ISURF,ISUB,AOV,POV,AEV,PEV,ERATIC,BLUAIR,REOAIR,BLUSUR,
24          3REOSUR,BLUSUB,REOSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
25          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
26    C
27    C     AIRCRAFT NAME DATA ARRAY
28    C
29           DATA NAMEAC/'HO','SE','LY','WA','SE','LA','WE','SU','AB',2*'MA',
30          1'TR','OR','VI','NI','AL','RM','A ','NX','SP','A ','MP','SS','PE',
31          1'-2','IL','Y ','AC','IO','KI','MR','IZ','ON','KI',2*'  ','SP',
32          1'S ','EX','R ','O4',2*'  ','KE','N ','NG','OO','E ','E ','NG',
33          12*'  ','RI','MK','  ','FR',3*'  ','R ',4*' ',4*'  ','TE',' I','  '
34          1,'EL',8*'  ',5*'  ','II','  ','ON',8*'  '/
35    C
36    C     AIRCRAFT DATA ARRAY
37    C
38           DATA ACDATA/11,12,15,10,12,16,12,14,10,24,31,15,30,2*35,24,3*4,3,
39          12*4,3,4,3*3,4*4,3,3*4,3,2*4,3,4,3*3,4*4,3,6,2*16,17,3*16,2*17,
40          12*29,4*16,17,9*1,4,5,1,3,1,2*3,2*29,9*0,5*29,3,1,9*0,3,4,3,2*4/
41    C
42    C      WEAPON NAME DATA ARRAY
43    C
44           DATA NAMEWP/2*'SS','SU','SS','21','16',2*'MK','TI',4*'MK','53',
45          1'AS',2*'MK','IK','MA','FR',2*'MB','LI','HE','37','30','ME','80',
46          1'OC',2*'N-','BR','N-',2*'  ',' 4',' 3','GE',' 2',' K',' L',' E',
47          1'3M','RO',2*' 4','AR','LA','AS',2*'U-','MB','DG',2*'5M','NO','F0',
48          1'  ','3 ','7 ','0C','14',2*'  ','8 ','7C','RF','0 ','2 ','5 ',
49          1'12','M ','C ','6 ','4 ','A ','F0','-1','25','45','0 ','EH',
50          12*'M ','N ','RS','  ',8*'  ','IS',9*'  ','N ','  ',2*'00','  ',
```

```
51          1'0G',3x'  ',' 3',' ',8x'  ','H ',11x'  ',2x'A ',5x'  ','75',' ',
52          127x'  ','MM',' '/
53      C
54      C       WEAPON DATA ARRAY
55      C
56           DATA WEPDA/5,4,2x6,2x5,6,5,6,5,6,3x5,6,5,4,6,5,6,2x4,2x3,4,4x3,
57          12x30,25,20,15,10,25,2x15,7,10,2x8,12,7,20,12,10,8,16,3,2,7x1,
58          12x1,2,16x3,2,9x4/
59      C
60      C       SHIP NAME ARRAYS
61      C
62           DATA NS1/'NI','AM','KI','F0','MI','EN','VI','CA','TA','BE','BA',
63          1'LE','AL','CH','L0','CL','AE','SP','MI','F0','C0','CH','CA','GE',
64          1'0L','KN','BA','GA','TA','LA','ET','G.','L0','NA','ST','PE','TU',
65          1'SK','SK','KI','M0','KA','KA','KY','KA','KA','K0','KA','SK','MI',
66          1'PE','RI','DE','YA','G0','H0','CH','VI','EC','N0','IN','BA','WE',
67          1'SH','AM','LE','R0','TA','LE','SA','BL','WH','RE','SW','VA','DA',
68          1'0B','TA','AN','ST','MA','RE','0B','ME','PE','DA','0.','RI','0B',
69          1'CL','JE','C0','M0','DU','T0','AC','SU','LA','TA','KE','CA','DE',
70          1'C0','SA','LE','RE','AG','DA','AR','NA','VI','AN','AU',2x'IM','LU'
71          1,'AL','DE','BE','CE','AL','SA','EX','T0','LU','HA','K0','TH',2x'U-
72          1'/
73      C
74           DATA NS2/'MI','ER','TT','RE','DW','TE','AG','LI','UX','LK','IN',
75          1'AH','BA','IC','NG','EV','GI','AU','TS','RA','0N','AR','AP','AR',
76          1'IV','0X','00','RC','ID','FE','HA',' W','S ','AW','UR','AM','LI',
77          1'IP','AT','EV','SK','RA','ES','ND','IV','NI','TL','SH','0R','AK',
78          1'TY','GA','LT','NK','LF','TE','AR','CT','H0','VE','VI','IS','AP',
79          1'EF','AZ','AN','TH','IB','0P','LI','AC','IT','S0','IF','LI','EA',
80          1'ER','IB','NA','. ','CK','ST','ER','LB','RT','RI','H.','VE','ER',
81          1'EM','AN','LB','NT','PE','UR','0N','FF',' G','RT','AS','SA','ST',
82          1'MM','V0',' B','D0','0S','PH','ET','AV','TT','DA','DA','PA','PE',
83          1'P0','PI',' C','AG','NT','BA','UA','-T','TI','TJ','MB','LN','ET',
84          1'20','2D'/
85      C
86           DATA NS3/'TZ','IC','Y ','ST','AY','AP','IN','F0','T0','NA','BA',
87          1'Y ','NT','AG',' B','EL','S ','AN','CH','ES','TZ','LE','EN','IN',
88          1'ER','  ','KE','IA','EN','YE','N ','AS','AN','HA','GE','IT','BE',
89          1'JA','E ','  ','VA','  ','TA','A ','AK','N ',2x'IN','Y ',2x'A ',
90          1'  ','A ','EE','  ','L ','LI','0R','  ','MB','NC','T0','0N',
91          1'FI','0N','DE','ES','AL','AR','3B','KW','BY','LU','TS','AN','0N',
92          1'0N','AL','P0','LA','ER','IN','0N','0U','H ','NG',' P','A ','0N',
93          1'EN','NE','ER','CA','AA','VI','IT','RE','AL','U ','AI','BL','IE',
94          1'. ','YA','0U','UB','TA','NE','HU','AL','0R','EA','CE','VI','TU',
95          1'  ','N0','HA','AM','AU','TA','0 ','AN','  ','EN','UR','  ','IS',
96          1'6 ','5 '/
97      C
98           DATA NS4/'  ','A ','HA','AL','  ','RI','IA','RN','N ','P ','ID',2x
99          1'  ','0 ','EA','AN','  ','CE','ER','T ','  ','S ','TE','G ',' H',3
100         1x'  ','T ','TT','AL','HI','GE','L ','0N','  ','E ','CK','  ',17x
```

```
      1'  ','  ',2x'  ','ER','IB','L ','  ','EL','  ','R ','AY','  ',
      1'D ','UR','OO','  ','TI','UR','T ','OU','  ','  ','LI','UR','SI',
      1'GO','  ','RN',2x'  ','ER',2x'  ','CE',' O','T ','LM','E ','LL',
      1'  ','N ','IS','  ','NT','AN','NN','RI','RD','LO','TA',2x'  ',
      1'SE','  ','IO',' D','  ','DO','OS',2x'  ','IS','IN','RO','OS',
      1'  ','G ','  ','S ','G ',4x'  '/
C
      DATA NS5/2x'  ','WK',2x'  ','SE','  ','IA',2x'  ','GE',3x'  ','CH'
     1,'D ',3x'  ','SH','  ','F.','R ','  ','. ',4x'  ','E ','LE','NG',
     1'LE',6x'  ',21x'  ','LE',2x'  ','D ',5x'  ','Y ','D ','  ','ON','E
     1 ','  ','GH','  ','  ','S ','EN','E ','UC','  ','E ',2x'  ','RY',
     12x'  ','AU','F ',3x'  ','E ',2x'  ','SO',2x'  ','CA','E ','VI',
     1'  ','NN','BL',4x'  ','V','OR',2x'  ','O ',2x'  ','TO',
     1'I ',11x'  '/
C
      DATA NS6/19x'  ','ER','  ',' A',2x'  ','PE',5x'  ','N ','TO','S ',
     16x'  ',21x'  ',15x'  ','T ','  ',2x'  ','T ','  ','HE',7x'  ',
     1'  ','AR',6x'  ','NN',4x'  ','ER','  ','AI','E ',4x'  ','EN','IA',
     15x'  ','FA',12x'  '/
C
      DATA NS7/19x'  ','MA','  ','DA',2x'  ','RR',6x'  ','N ',7x'  ',
     121x'  ',17x'  ',12x'  ','  ','C ',6x'  ','IE',4x'  ','E ','  ',
     1'S ','  ',4x'  ','ET',6x'  ','RO',12x'  '/
C
      DATA NS8/19x'  ','N ','  ','MS',2x'  ','Y ',14x'  ',21x'  ',
     117x'  ',12x'  ',8x'  ','RE',12x'  ','O ',19x'  '/
C
C      SHIP DATA ARRAYS
C
      DATA S1/9x3,4,3,4,3,3,4,7x3,4,4,3,3x2,3,3x2,4x3,3x2,4x3,5x4,5x3,
     13x2,3,4,2x2,12x3,2,3x3,2,5x3,2,2,4,3x3,2,12x3,9x2,2x3,3x4,3,3x2,
     13,4x2,4,4,3,3,2,2/
C
      DATA S2/0,3,4x0,4x5,5x4,0,2x5,2x3,2x4,2x3,4x5,4x0,5,4x4,2x3,3x4,
     12x3,4,7x3,2x4,2x3,2x4,2x3,2x3,5,2x4,7x3,2x4,3x3,5x4,3,0,4,3,5,3,3,
     14,3,0,6x5,2x4,5,4,3,4,3,3x4,2x0,2x3,4,3,2,4,4x3,0,3x3,4,3x3,2x0/
C
      DATA S3/0,4,4x0,5,8x4,3,2x5,6x4,5,3x4,9x5,2x4,3x5,2x4,5,2x4,5,4x4,
     12x5,2x4,2x5,2x4,2x4,5,13x4,3,5x4,3,0,2x4,5,4,3,2x4,0,6x5,8x4,2x5,
     112x4,0,3x4,6x4/
C
      DATA S4/6x0,4x1,6x0,2x1,6x0,4x1,4x0,5x1,2x0,3x1,2x0,1,2x0,1,12x0,
     117x0,1,5x0,3x0,1,2x0,1,2x0,6x1,2x0,2x1,3x0,3x1,2x0,14x0,6x0/
C
      DATA S5/6x0,4x1,6x0,2x1,6x0,13x1,2x0,3x1,2x0,1,2x0,1,12x0,17x0,
     11,5x0,3x0,1,2x0,3x0,6x1,2x0,1,9x0,6x0,2x1,6x0,6x0/
C
      DATA S6/2,4x1,2,3x4,6x3,2,12x4,5,3x4,3x5,4,5,2x2,2,4x3,9x4,3,5,3,
     12x4,2x2,2x3,14x4,5,5x4,5,2,4x4,5,2,2x3,14x4,4x5,2x3,9x4,3x5,
     14x4,2x5/
```

```
151     C
152           DATA S7/6*2,4*3,2,4*3,1,2*3,2*2,3,7*2,3,3*2,5*3,2*2,4*3,5*2,2*1,
153          12,1,2*3,0,2,2*3,2*2,17*2,6*2,1,2,1,2,1,2,2*2,1,6*3,2*2,3,4*2,3*3,
154          12*0,2*2,3,2,1,5*2,0,2,2*1,3*2,3*1/
155     C
156           DATA S8/6*0,9*15,0,15,16,6*15,16,3*15,4*7,7*3,4,20,2*4,21,4,3*21,
157          129,3*21,4*5,2,5,1,5,0,18,3*16,7*23,5*9,5*16,9,0,18,23,16,18,9,3*0,
158          15*12,4*11,12,3*11,5*13,11*17,3*14,17,3*14,2*8/
159     C
160           DATA S9/6*0,4,2*5,3,5*5,0,5,4,6*5,4,3*5,4*4,7*3,5,3,2*4,6,4,3*6,
161          14,2*7,8,4*4,1,4,1,3,0,3,3*4,7*6,5*4,6*4,0,3*4,3,4,3*0,5*5,8*4,5*5,
162          112*5,4,5,5,3*4,2*5/
163     C
164           DATA S10/6*0,9*16,2*0,15,6*16,0,3*16,4*0,7*7,4*21,0,21,0,29,22,0,
165          16,0,29,4*0,5,0,5,0,0,23,3*0,16,6*0,5*10,0,4*23,10,4*0,23,10,5*0,
166          14*19,2*25,19,25,2*26,25,5*0,4*0,27,0,2*27,0,27,24,3*0,15,3*28,2*0/
167     C
168           DATA S11/6*0,4,2*3,6*4,2*0,5,6*4,0,3*4,4*0,7*4,4*6,0,6,0,4,6,0,4,
169          10,4,4*0,4,0,4,0,0,6,3*0,4,6*0,5*4,0,2*4,2*5,4,4*0,2*4,5*0,4*4,2*6,
170          14,4*6,5*4,4*0,6,0,5,6,0,2*6,3*0,5,3*6,0,0/
171     C
172           DATA S12/39*0,2*5,2*29,4*0,29,12*0,17*0,6*0,6*0,6*0,26,4*0,25,2*0,
173          129,6*0,14*0,6*0/
174     C
175           DATA S13/39*0,4*4,4*0,4,12*0,17*0,6*0,6*0,6*0,6,4*0,6,2*0,4,6*0,
176          114*0,6*0/
177     C
178           DATA S14/6*2,5*5,2*0,2*5,0,6,5,6*0,4*5,11*0,4*1,17*0,2,4,3*3,3*4,
179          19*0,2*2,4*0,2,2*0,2,2*0,16,8,0,3*3,2*0,3,12*0,2*9,2,9,2*0,9,7*0,
180          16*0/
181     C
182           DATA S15/6*6,3*2,2*1,2*0,2*1,0,2,1,6*0,2,3*1,11*0,10,5,2*1,17*0,
183          110,1,2,5*1,9*0,2,2*1,3*0,5,2*0,1,2*0,10,6,0,3*1,2*0,1,2*0,
184          110*0,9,4,1,1,2*0,2*2,11*0/
185     C
186           DATA S16/6*14,33*0,21*0,17*0,6*0,6*0,8,20*0,14*0,6*0/
187     C
188           DATA S17/6*6,33*0,21*0,17*0,6*0,6*0,5,20*0,14*0,6*0/
189     C
190           DATA S18/6*3,10*2,12*1,11*2,3,4*2,8*1,8*2,2*2,10*1,5*2,5*1,2,3,
191          14*1,2,3,2*2,13*1,5*2,2*2,9*1,3*2,4*1,2*2/
192           DATA MAXPLT/100/
193           DATA ALPHA/'Y','N'/
194     C
195     C     EQUALIZE WEPOA AND WEP SO WEP CAN BE COMMONED
196     C
197           DO 81 I=1,29
198           DO 80 J=1,3
199           WEP(I,J)=WEPOA(I,J)
200     80    CONTINUE
```

```
81    CONTINUE
C
C     INITIALIZE THE GAME PARAMETERS
C
99    TURN=0
      BLUAIR=0
      BLUSUR=0
      BLUSUB=0
      REDAIR=0
      REDSUR=0
      REDSUB=0
C
C     INITIALIZE THE ARRAY PLATFM AND OTHERS
C
      DO 83 I=1,MAXPLT
      DO 82 J=1,37
      PLATFM(I,J)=0
82    CONTINUE
      WATER(I)=-1
      DETECT(I,1)=-1
      HIT(I,1)=-1
      SUNK(I)=-1
      BUFFER(I,1)=0
      BUFFER(I,2)=0
      BUFFER(I,3)=0
      BUFFER(I,4)=0
      BUFFER(I,5)=0
      BUFFER(I,6)=1
      BUFFER(I,7)=1
      BUFFER(I,8)=1
83    CONTINUE
C
C     MENU
C
1     WRITE(6,100)
100   FORMAT('1THE FOLLOWING OPTIONS ARE AVAILABLE TO YOU:',
     1///,' 1. ZERO ANY PREVIOUSLY ESTABLISHED DATA BASE.',
     1//,' 2. EXIT THE GAME.',
     1//,' 3. INITIALIZE GAME PARAMETERS AND COMMENCE PLAY.',
     1///,' ENTER AN OPTION.')
      READ(5,200) INPUT
200   FORMAT(70A1)
      CALL NONVRT(INPUT,INDEX)
      IF(INDEX.NE.1) GO TO 2
201   FORMAT(A1)
C
      WRITE(6,111)
111   FORMAT('1THE OPTION YOU HAVE SELECTED WILL DESTROY ANY FILES THAT'
     1,/,' YOU HAVE ALREADY CREATED.  IF YOU CONTINUE, YOU WILL HAVE',
     1/,' TO RELOAD ALL USER INPUTS.  TYPE Y(ES) TO ERASE THE FILES.',
```

```
     1/,' OR TYPE N(O) IF YOU WANT TO RETURN TO MENU SELECTION WITHOUT',
     1/,' ERASING THE FILES.')
      READ(5,200) INPUT
      CALL INVRT1(INPUT,A)
      IF(A.NE.ALPHA(1)) GO TO 1
      GO TO 99
2     IF(INDEX.EQ.2) GO TO 9999
C
C     LOAD GAME PARAMETERS
C
4     HALF=MAXPLT
      HALF=HALF/2
      WRITE(6,120) MAXPLT,HALF
120   FORMAT('1YOU ARE NOW GOING TO ENTER THE BASIC GAME PARAMETERS.',
     1/,' ONE WORD OF CAUTION.  THE PRESENT GAME CAN ONLY HANDLE A',
     1/,' TOTAL OF ',I4,' WEAPON SYSTEMS.  INCLUDED IN THESE SYSTEMS',
     1/,' ARE SURFACE SHIPS, SUBMARINES AND AIRCRAFT.  THE NUMBER OF',
     1/,' AIRCRAFT ARE DETERMINED BY THE SHIP CLASS AS REFLECTED IN THE'
     1,/,' DATA BASE, AND WILL BE LOADED AUTOMATICALLY WHEN YOU SELECT',
     1/,' THE SHIP CLASS.  PRIOR TO BEGINNING THE LOADING PROCEDURE',
     1/,' YOU SHOULD ENSURE THAT THE TOTAL NUMBER OF EACH PLAYERS',
     1/,' WEAPON SYSTEMS DOES NOT EXCEED ',I4,'.')
      WRITE(6,121)
121   FORMAT(////,' ENTER THE NUMBER OF GAME TURNS YOU WANT TO PLAY.')
      READ(5,200) INPUT
      CALL NONVRT(INPUT,TURN)
166   WRITE(6,167)
167   FORMAT(//,'0ENTER THE WATER CONDITION (0-NORMAL, 1-ERRATIC).')
      READ(5,200) INPUT
      CALL NONVRT(INPUT,ERATIC)
      IF(ERATIC.EQ.0.OR.ERATIC.EQ.1) GO TO 5
      WRITE(6,168)
168   FORMAT(/,' YOU HAVE MADE AN ERROR ENTERING THE WATER CONDITION.')
      GO TO 166
5     WRITE(6,122)
122   FORMAT('1THE FOLLOWING INPUTS APPLY ONLY TO THE BLUE PLAYER',
     1/,' THE FIRST IS THE BLUE PLAYERS PASSWORD WHICH IS',
     1/,' A MAXIMUM OF 6 CHARACTERS.  THE PASSWORD IS IMPORTANT BECAUSE'
     1,/,' IT IS REQUIRED PRIOR TO THE INPUT OF COMBAT, MOVE AND SEARCH'
     1,/,' COMMANDS.  IF YOU FORGET THE PASSWORD, YOU WILL HAVE TO RETUR
     1N',/,' TO MENU SELECTION, AND RESTART THE GAME.',///,' SELECT AND
     1ENTER YOUR PASSWORD')
      READ(5,200) INPUT
      CALL INVRT6(INPUT,BLUPAS)
      WRITE(6,123) BLUPAS
123   FORMAT(//,' YOUR PASSWORD WILL BE -',6A1,'- .  IF YOU ARE',
     1/,' READY TO CONTINUE TYPE Y(ES).  TYPE N(O) IF YOU WISH TO',
     1/,' CHANGE YOUR PASSWORD.')
      READ(5,200) INPUT
      CALL INVRT1(INPUT,A)
```

```
      IF(A.EQ.ALPHA(2)) GO TO 5
      WRITE(6,124)
124   FORMAT(//,'0ENTER THE NUMBER OF BLUE SURFACE SHIPS.')
      READ(5,200) INPUT
      CALL NONVRT(INPUT,BLUSUR)
      WRITE(6,126)
126   FORMAT(//,'0ENTER THE NUMBER OF BLUE SUBMARINES.')
      READ(5,200)  INPUT
      CALL NONVRT(INPUT,BLUSUB)
      WRITE(6,127)
127   FORMAT(//,'0ENTER THE NUMBER OF BLUE SHIPS THAT MUST',
     1/,' BE DESTROYED FOR PREMATURE GAME ENDING.')
      READ(5,200)  INPUT
      CALL NONVRT(INPUT,BLUDES)
      WRITE(6,128)  BLUSUR,BLUSUB,BLUDES
128   FORMAT(//,' ECHO',//,' BLUE SURFACE SHIPS - ',I4,
     1/,' BLUE SUBMARINES - ',I4,
     1/,' NUMBER OF BLUE SHIPS DESTROYED TO END GAME - ',I4)
      WRITE(6,129)
129   FORMAT(//,'0TYPE Y(ES) IF THESE VALUES ARE CORRECT, OR N(O) IF',
     1/,' YOU WANT TO REENTER THEM.')
      READ(5,200)  INPUT
      CALL INVRT1(INPUT,A)
      IF(A.EQ.ALPHA(2)) GO TO 5
6     WRITE(6,130)
130   FORMAT('1THE FOLLOWING INPUTS APPLY ONLY TO THE RED PLAYER',
     1/,' THE FIRST IS THE RED PLAYERS PASSWORD WHICH IS A ',
     1/,' MAXIMUM OF 6 CHARACTERS.  THE PASSWORD IS IMPORTANT BECAUSE',
     1/,' IT IS REQUIRED PRIOR TO THE INPUT OF COMBAT, MOVE AND SEARCH'
     1,/,' COMMANDS.  IF YOU FORGET THE PASSWORD, YOU WILL HAVE TO RETUR
     1N',/,' TO MENU SELECTION, AND RESTART THE GAME.',///,' SELECT AND
     1ENTER YOUR PASSWORD.')
      READ(5,200)  INPUT
      CALL INVRT6(INPUT,REDPAS)
      WRITE(6,131)  REDPAS
131   FORMAT(//,' YOUR PASSWORD WILL BE -',6A1,'- . IF YOU ARE',
     1/,' READY TO CONTINUE TYPE Y(ES).  TYPE N(O) IF YOU WISH TO',
     1/,' CHANGE YOUR PASSWORD.')
      READ(5,200)  INPUT
      CALL INVRT1(INPUT,A)
      IF(A.EQ.ALPHA(2)) GO TO 6
      WRITE(6,132)
132   FORMAT(//,'0ENTER THE NUMBER OF RED SURFACE SHIPS.')
      READ(5,200)  INPUT
      CALL NONVRT(INPUT,REDSUR)
      WRITE(6,135)
135   FORMAT(//,'0ENTER THE NUMBER OF RED SUBMARINES.')
      READ(5,200)  INPUT
      CALL NONVRT(INPUT,REDSUB)
      WRITE(6,136)
```

```
351     136   FORMAT(//,'0ENTER THE NUMBER OF RED SHIPS THAT MUST',
352           1/,' BE DESTROYED FOR PREMATURE GAME ENDING.')
353           READ(5,200) INPUT
354           CALL NONVRT(INPUT,REDDES)
355           WRITE(6,137) REDSUR,REDSUB,REDDES
356     137   FORMAT(//,' ECHO',//,' RED SURFACE SHIPS - ',I4,
357           1/,' RED SUBMARINES - ',I4,
358           1/,' NUMBER OF RED SHIPS DESTROYED TO END GAME - ',I4)
359           WRITE(6,138)
360     138   FORMAT(//,'0TYPE Y(ES) IF THESE VALUES ARE CORRECT, OR N(O) IF',
361           1/,' YOU WANT TO REENTER THEM.')
362           READ(5,200) INPUT
363           CALL INVRT1(INPUT,A)
364           IF(A.EQ.ALPHA(2))GO TO 6
365     C
366     C     SET VARIABLES TO BE USED AS POINTERS TO THE PLATFM ARRAY
367     C
368           PRSUB2=MAXPLT
369           PRSUB1=MAXPLT-REDSUB+1
370           PBSUB2=PRSUB1-1
371           PBSUB1=PBSUB2-BLUSUB+1
372           PRSUR2=PBSUB1-1
373           PRSUR1=PRSUR2-REDSUR+1
374           PBSUR2=PRSUR1-1
375           PBSUR1=PBSUR2-BLUSUR+1
376     C
377     C     LOAD BLUE WEAPON SYSTEMS
378     C
379     7     WRITE(6,140)
380     140   FORMAT('1YOU ARE NOW READY TO LOAD THE BLUE WEAPON SYSTEMS.',
381           1//,' ENTER THE BLUE PASSWORD.')
382           READ(5,200) INPUT
383           CALL INVRT6(INPUT,CHECK)
384           A=0
385           DO 141 I=1,6
386           IF(CHECK(I).NE.BLUPAS(I)) A=1
387           IF(A.EQ.1) GO TO 142
388     141   CONTINUE
389     142   IF(A.NE.1) GO TO 8
390           WRITE(6,143)
391     143   FORMAT(//,' YOU HAVE MADE AN ERROR IN YOUR PASSWORD.  TYPE Y(ES) ',
392           1/,' IF YOU WANT TO TRY AGAIN.  OTHERWISE YOU WILL BE',
393           1/,' RETURNED TO MENU SELECTION.')
394           READ(5,200) INPUT
395           CALL INVRT1(INPUT,A)
396           IF(A.EQ.ALPHA(1)) GO TO 7
397           GO TO 1
398     8     SHIPNO=0
399           BLUAIR=0
400           IF(BLUSUR.EQ.0) GO TO 170
```

```
401           DO 165 SHIP=PBSUR1,PBSUR2
402           SHIPNO=SHIPNO+1
403           WRITE(6,144) SHIPNO
404     144   FORMAT('1BLUE SURFACE SHIP NUMBER ',I4,//,' ENTER A 6 CHARACTER',
405          1' NAME FOR THE SHIP.')
406           READ(5,200) INPUT
407           CALL INVRT6(INPUT,CHECK)
408           DO 145 J=1,6
409           NAME(SHIP,J)=CHECK(J)
410     145   CONTINUE
411     147   WRITE(6,146) CHECK
412     146   FORMAT('0ENTER THE SHIP CLASS OF ',6A1)
413           READ(5,202) CLASNA
414     202   FORMAT(8A2)
415           CALL SCLASS(CLASNA,CLASIN,NAMESH)
416           IF(CLASIN.EQ.0) WRITE(6,148)
417     148   FORMAT('0YOU HAVE MADE A MISTAKE.  ENTER THE SHIP CLASS WITH NO',
418          1/,'  LEADING BLANKS USING A PROPER CLASS NAME.')
419           IF(CLASIN.EQ.0) GO TO 147
420           PLATFM(SHIP,30)=CLASIN
421           WRITE(6,149) CHECK
422     149   FORMAT('0ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL ',
423          1/,' LOCATION OF ',6A1)
424           READ(5,200) INPUT
425           CALL NONVRT(INPUT,A)
426           PLATFM(SHIP,3)=A
427           PLATFM(SHIP,8)=1
428           PLATFM(SHIP,20)=SHIPDA(CLASIN,18)
429           PLATFM(SHIP,12)=SHIPDA(CLASIN,1)
430           PLATFM(SHIP,13)=SHIPDA(CLASIN,2)
431           PLATFM(SHIP,14)=SHIPDA(CLASIN,3)
432           PLATFM(SHIP,15)=SHIPDA(CLASIN,9)
433           PLATFM(SHIP,16)=SHIPDA(CLASIN,11)
434           PLATFM(SHIP,17)=SHIPDA(CLASIN,13)
435           PLATFM(SHIP,18)=SHIPDA(CLASIN,6)
436           PLATFM(SHIP,19)=SHIPDA(CLASIN,7)
437           PLATFM(SHIP,21)=SHIPDA(CLASIN,8)
438           PLATFM(SHIP,22)=SHIPDA(CLASIN,10)
439           PLATFM(SHIP,23)=SHIPDA(CLASIN,12)
440           PLATFM(SHIP,24)=SHIPDA(CLASIN,4)
441           PLATFM(SHIP,25)=SHIPDA(CLASIN,5)
442           PLATFM(SHIP,26)=SHIPDA(CLASIN,14)
443           PLATFM(SHIP,27)=SHIPDA(CLASIN,16)
444           PLATFM(SHIP,28)=SHIPDA(CLASIN,15)
445           PLATFM(SHIP,29)=SHIPDA(CLASIN,17)
446     C
447     C
448     C     LOAD BLUE WEAPON SYSTEMS - AIRCRAFT
449     C
450     150   DO 155 I=26,27
```

```
451           IF(PLATFM(SHIP,1).EQ.0) GO TO 165
452           ACINDX=PLATFM(SHIP,1)
453           WRITE(6,151) CLASNA, PLATFM(SHIP,1+2),(NAMEAC(ACINDX,J),J=1,6)
454     151   FORMAT(//,'0THE ',8A2,' CLASS OF SHIP HAS ',I2,1X,6A2,1X,'TYPE',
455          1' AIRCRAFT.')
456           PLATFM(SHIP,1+5)=BLUAIR+1
457           BLUAIR=BLUAIR+PLATFM(SHIP,1+2)
458           PLATFM(SHIP,1+7)=BLUAIR
459           ACOUNT=0
460           P=PLATFM(SHIP,1+5)
461           Q=PLATFM(SHIP,1+7)
462           DO 154 AC=P,Q
463           ACOUNT=ACOUNT+1
464           WRITE(6,152) ACOUNT
465     152   FORMAT('0ENTER A 6 CHARACTER NAME FOR AIRCRAFT ',I2)
466           READ(5,200) INPUT
467           CALL INVRT6(INPUT,CHECK)
468           DO 153 J=1,6
469           NAME(AC,J)=CHECK(J)
470     153   CONTINUE
471           PLATFM(AC,3)=PLATFM(SHIP,3)
472           PLATFM(AC,8)=0
473           PLATFM(AC,12)=ACDATA(ACINDX,1)
474           PLATFM(AC,13)=ACDATA(ACINDX,2)
475           PLATFM(AC,14)=ACDATA(ACINDX,3)
476           PLATFM(AC,15)=ACDATA(ACINDX,5)
477           PLATFM(AC,16)=ACDATA(ACINDX,7)
478           PLATFM(AC,17)=ACDATA(ACINDX,4)
479           PLATFM(AC,18)=ACDATA(ACINDX,6)
480           PLATFM(AC,20)=1
481           PLATFM(AC,21)=SHIP
482           PLATFM(AC,22)=PLATFM(SHIP,1)
483     154   CONTINUE
484     155   CONTINUE
485     165   CONTINUE
486     C
487     C     LOAD BLUE WEAPON SYSTEMS - SUBMARINES
488     C
489     170   IF(BLUSUB.EQ.0) GO TO 10
490           SUBNO=0
491           DO 175 SUB=PBSUB1,PBSUB2
492           SUBNO=SUBNO+1
493           WRITE(6,171) SUBNO
494     171   FORMAT('1BLUE SUBMARINE NUMBER ',I4,//,' ENTER A 6 DIGIT',
495          1' CHARACTER NAME FOR THE SUBMARINE.')
496           READ(5,200) INPUT
497           CALL INVRT6(INPUT,CHECK)
498           DO 172 J=1,6
499           NAME(SUB,J)=CHECK(J)
500     172   CONTINUE
```

```
501     174   WRITE(6,173) CHECK
502     173   FORMAT('0ENTER THE SUBMARINE CLASS OF ',6A1)
503           READ(5,202) CLASNA
504           CALL SCLASS(CLASNA,CLASIN,NAMESH)
505           IF(CLASIN.EQ.0) WRITE(6,148)
506           IF(CLASIN.EQ.0) GO TO 174
507           PLATFM(SUB,30)=CLASIN
508           WRITE(6,176) CHECK
509     176   FORMAT('0ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL ',
510          1/,' LOCATION OF ',6A1)
511           READ(5,200) INPUT
512           CALL NONVRT(INPUT,A)
513           PLATFM(SUB,3)=A
514           PLATFM(SUB,8)=2
515           PLATFM(SUB,20)=SHIPDA(CLASIN,18)
516           PLATFM(SUB,12)=SHIPDA(CLASIN,1)
517           PLATFM(SUB,13)=SHIPDA(CLASIN,2)
518           PLATFM(SUB,14)=SHIPDA(CLASIN,3)
519           PLATFM(SUB,15)=SHIPDA(CLASIN,9)
520           PLATFM(SUB,16)=SHIPDA(CLASIN,11)
521           PLATFM(SUB,17)=SHIPDA(CLASIN,13)
522           PLATFM(SUB,18)=SHIPDA(CLASIN,6)
523           PLATFM(SUB,19)=SHIPDA(CLASIN,7)
524           PLATFM(SUB,21)=SHIPDA(CLASIN,8)
525           PLATFM(SUB,22)=SHIPDA(CLASIN,10)
526           PLATFM(SUB,23)=SHIPDA(CLASIN,12)
527           PLATFM(SUB,24)=SHIPDA(CLASIN,4)
528           PLATFM(SUB,25)=SHIPDA(CLASIN,5)
529           PLATFM(SUB,26)=SHIPDA(CLASIN,14)
530           PLATFM(SUB,27)=SHIPDA(CLASIN,16)
531           PLATFM(SUB,28)=SHIPDA(CLASIN,15)
532           PLATFM(SUB,29)=SHIPDA(CLASIN,17)
533     175   CONTINUE
534     C
535     C
536     C     LOAD RED WEAPON SYSTEMS
537     C
538     10    WRITE(6,240)
539     240   FORMAT('1YOU ARE NOW READY TO LOAD THE RED WEAPON SYSTEMS.',
540          1//,' ENTER THE RED PASSWORD.')
541           READ(5,200) INPUT
542           CALL INVRT6(INPUT,CHECK)
543           A=0
544           DO 241 I=1,6
545           IF(CHECK(I).NE.REDPAS(I)) A=1
546           IF(A.EQ.1) GO TO 242
547     241   CONTINUE
548     242   IF(A.NE.1) GO TO 11
549           WRITE(6,243)
550     243   FORMAT(//,' YOU HAVE MADE AN ERROR IN YOUR PASSWORD.  TYPE Y(ES) ',
```

```
     1/,' IF YOU WANT TO TRY AGAIN.  OTHERWISE YOU WILL BE',
     1/,' RETURNED TO MENU SELECTION.')
      READ(5,200) INPUT
      CALL INVRT1(INPUT,A)
      IF(A.EQ.ALPHA(1)) GO TO 10
      GO TO 1
11    REQAIR=BLUAIR
      IF(REDSUR.EQ.0) GO TO 270
      SHIPNO=0
      DO 265 SHIP=PRSUR1,PRSUR2
      SHIPNO=SHIPNO+1
      WRITE(6,244) SHIPNO
244   FORMAT('1REQ SURFACE SHIP NUMBER ',I4,//,' ENTER A 6 CHARACTER',
     1' NAME FOR THE SHIP.')
      READ(5,200) INPUT
      CALL INVRT6(INPUT,CHECK)
      DO 245 J=1,6
      NAME(SHIP,J)=CHECK(J)
245   CONTINUE
247   WRITE(6,246) CHECK
246   FORMAT('0ENTER THE SHIP CLASS OF ',6A1)
      READ(5,203) CLASNA
203   FORMAT(8A2)
      CALL SCLASS(CLASNA,CLASIN,NAMESH)
      IF(CLASIN.EQ.0) WRITE(6,248)
248   FORMAT('0YOU HAVE MADE A MISTAKE.  ENTER THE SHIP CLASS WITH NO',
     1/,'  LEADING BLANKS USING A PROPER CLASS NAME.')
      IF(CLASIN.EQ.0) GO TO 247
      PLATFM(SHIP,30)=CLASIN
      WRITE(6,249) CHECK
249   FORMAT('0ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL ',
     1/,' LOCATION OF ',6A1)
      READ(5,200) INPUT
      CALL NONVRT(INPUT,A)
      PLATFM(SHIP,3)=A
      PLATFM(SHIP,8)=1
      PLATFM(SHIP,20)=SHIPDA(CLASIN,18)
      PLATFM(SHIP,12)=SHIPDA(CLASIN,1)
      PLATFM(SHIP,13)=SHIPDA(CLASIN,2)
      PLATFM(SHIP,14)=SHIPOA(CLASIN,3)
      PLATFM(SHIP,15)=SHIPDA(CLASIN,9)
      PLATFM(SHIP,16)=SHIPOA(CLASIN,11)
      PLATFM(SHIP,17)=SHIPOA(CLASIN,13)
      PLATFM(SHIP,18)=SHIPOA(CLASIN,6)
      PLATFM(SHIP,19)=SHIPOA(CLASIN,7)
      PLATFM(SHIP,21)=SHIPDA(CLASIN,8)
      PLATFM(SHIP,22)=SHIPOA(CLASIN,10)
      PLATFM(SHIP,23)=SHIPOA(CLASIN,12)
      PLATFM(SHIP,24)=SHIPDA(CLASIN,4)
      PLATFM(SHIP,25)=SHIPDA(CLASIN,5)
```

```
601           PLATFM(SHIP,26)=SHIPDA(CLASIN,14)
602           PLATFM(SHIP,27)=SHIPDA(CLASIN,16)
603           PLATFM(SHIP,28)=SHIPDA(CLASIN,15)
604           PLATFM(SHIP,29)=SHIPDA(CLASIN,17)
605     C
606     C
607     C     LOAD RED WEAPON SYSTEMS - AIRCRAFT
608     C
609     250   DO 255 I=26,27
610           IF(PLATFM(SHIP,I).EQ.0) GO TO 265
611           ACINDX=PLATFM(SHIP,I)
612           WRITE(6,251) CLASNA, PLATFM(SHIP,I+2),(NAMEAC(ACINDX,J),J=1,6)
613     251   FORMAT(//,'OTHE ',8A2,' CLASS OF SHIP HAS ',I2,1X,6A2,1X,'TYPE',
614          1' AIRCRAFT.')
615           PLATFM(SHIP,I+5)=REDAIR+1
616           REDAIR=REDAIR+PLATFM(SHIP,I+2)
617           PLATFM(SHIP,I+7)=REDAIR
618           ACOUNT=0
619           P=PLATFM(SHIP,I+5)
620           Q=PLATFM(SHIP,I+7)
621           DO 254 AC=P,Q
622           ACOUNT=ACOUNT+1
623           WRITE(6,252) ACOUNT
624     252   FORMAT('0ENTER A 6 CHARACTER NAME FOR AIRCRAFT ',I2)
625           READ(5,200) INPUT
626           CALL INVAT6(INPUT,CHECK)
627           DO 253 J=1,6
628           NAME(AC,J)=CHECK(J)
629     253   CONTINUE
630           PLATFM(AC,3)=PLATFM(SHIP,3)
631           PLATFM(AC,8)=0
632           PLATFM(AC,12)=ACDATA(ACINDX,1)
633           PLATFM(AC,13)=ACDATA(ACINDX,2)
634           PLATFM(AC,14)=ACDATA(ACINDX,3)
635           PLATFM(AC,15)=ACDATA(ACINDX,5)
636           PLATFM(AC,16)=ACDATA(ACINDX,7)
637           PLATFM(AC,17)=ACDATA(ACINDX,4)
638           PLATFM(AC,18)=ACDATA(ACINDX,6)
639           PLATFM(AC,20)=1
640           PLATFM(AC,21)=SHIP
641           PLATFM(AC,22)=PLATFM(SHIP,I)
642     254   CONTINUE
643     255   CONTINUE
644     265   CONTINUE
645     C
646     C     LOAD RED WEAPON SYSTEMS - SUBMARINES
647     C
648     270   IF(REDSUB.EQ.0) GO TO 15
649           SUBNO=0
650           DO 275 SUB=PRSUB1,PRSUB2
```

```
651           SUBNO=SUBNO+1
652           WRITE(6,271) SUBNO
653     271   FORMAT('1RED SUBMARINE NUMBER ',I4,//,' ENTER A 6 DIGIT',
654          1' CHARACTER NAME FOR THE SUBMARINE.')
655           READ(5,200) INPUT
656           CALL INVRT6(INPUT,CHECK)
657           DO 272 J=1,6
658           NAME(SUB,J)=CHECK(J)
659     272   CONTINUE
660     274   WRITE(6,273) CHECK
661     273   FORMAT('0ENTER THE SUBMARINE CLASS OF ',6A1)
662           READ(5,203) CLASNA
663           CALL SCLASS(CLASNA,CLASIN,NAMESH)
664           IF(CLASIN.EQ.0) WRITE(6,248)
665           IF(CLASIN.EQ.0) GO TO 274
666           PLATFM(SUB,30)=CLASIN
667           WRITE(6,276) CHECK
668     276   FORMAT('0ENTER THE 4 DIGIT HEX NUMBER OF THE INITIAL ',
669          1/,' LOCATION OF ',6A1)
670           READ(5,200) INPUT
671           CALL NONVRT(INPUT,A)
672           PLATFM(SUB,3)=A
673           PLATFM(SUB,8)=2
674           PLATFM(SUB,20)=SHIPDA(CLASIN,18)
675           PLATFM(SUB,12)=SHIPDA(CLASIN,1)
676           PLATFM(SUB,13)=SHIPDA(CLASIN,2)
677           PLATFM(SUB,14)=SHIPDA(CLASIN,3)
678           PLATFM(SUB,15)=SHIPDA(CLASIN,9)
679           PLATFM(SUB,16)=SHIPDA(CLASIN,11)
680           PLATFM(SUB,17)=SHIPDA(CLASIN,13)
681           PLATFM(SUB,18)=SHIPDA(CLASIN,6)
682           PLATFM(SUB,19)=SHIPDA(CLASIN,7)
683           PLATFM(SUB,21)=SHIPDA(CLASIN,8)
684           PLATFM(SUB,22)=SHIPDA(CLASIN,10)
685           PLATFM(SUB,23)=SHIPDA(CLASIN,12)
686           PLATFM(SUB,24)=SHIPDA(CLASIN,4)
687           PLATFM(SUB,25)=SHIPDA(CLASIN,5)
688           PLATFM(SUB,26)=SHIPDA(CLASIN,14)
689           PLATFM(SUB,27)=SHIPDA(CLASIN,16)
690           PLATFM(SUB,28)=SHIPDA(CLASIN,15)
691           PLATFM(SUB,29)=SHIPDA(CLASIN,17)
692     275   CONTINUE
693     C
694     C     SET VARIABLES TO BE USED AS POINTERS TO THE PLATFORM ARRAY
695     C
696     15    PBAIR1=1
697           PBAIR2=BLUAIR
698           REDAIR=REDAIR-BLUAIR
699           PRAIR1=BLUAIR+1
700           PRAIR2=PRAIR1+REDAIR-1
```

```
      IAIR=BLUAIR+REDAIR
      ISURF=BLUSUR+REDSUR
      ISUB=BLUSUB+REDSUB
      IPLAT=IAIR+ISURF+ISUB
      IF (IPLAT.LE.MAXPLT) GO TO 16
      WRITE (6,280)
280   FORMAT ('0THE TOTAL NUMBER OF WEAPON SYSTEMS HAS EXCEEDED THE MAXIM
     1UM',/,' NUMBER OF PLATFORMS ALLOWED.  YOU WILL HAVE',
     1/,' TO RESTART THE GAME.')
      GO TO 99
C
C     LOOP TO CLOSE PLATFM AND REASSIGN POINTERS
C
16     IF (PBSUR1.EQ. (PRAIR2+1)) GO TO 17
      ISHIP=ISURF+ISUB
      DO 283 I=1,ISHIP
      DO 281 J=1,34
      PLATFM (PRAIR2+I,J) =PLATFM (PBSUR1+I-1,J)
281   CONTINUE
      DO 282 J=1,6
      NAME (PRAIR2+I,J) =NAME (PBSUR1+I-1,J)
282   CONTINUE
      DO 284 J=28,29
      IF (PLATFM (PRAIR2+I,J) .EQ.0) GO TO 283
      P=PLATFM (PRAIR2+I,J+3)
      Q=PLATFM (PRAIR2+I,J+5)
      DO 285 K=P,Q
      PLATFM (K,21) =PRAIR2+I
285   CONTINUE
284   CONTINUE
283   CONTINUE
      DO 286 I=1,IPLAT
      BUFFER (I,4) =PLATFM (I,3)
286   CONTINUE
      PBSUR1=PRAIR2+1
      PBSUR2=PBSUR1+BLUSUR-1
      PRSUR1=PBSUR2+1
      PRSUR2=PRSUR1+REDSUR-1
      PBSUB1=PRSUR2+1
      PBSUB2=PBSUB1+BLUSUB-1
      PRSUB1=PBSUB2+1
      PRSUB2=PRSUB1+REDSUB-1
C
C     CONTROL OF GAME TURNS
C
17    ITURN=0
19    WRITE (6,300)
300   FORMAT ('1BLUE CALL TO STATUS ROUTINES.  ENTER THE BLUE PASSWORD.')
      READ (5,200)  INPUT
      CALL INVRTS (INPUT,CHECK)
```

```
751           CALL STATUS(CHECK,ERROR,NAMEWP)
752           IF(ERROR.EQ.0) GO TO 20
753           WRITE(6,301)
754     301   FORMAT('0TYPE Y(ES) TO TRY THE STATUS CALL AGAIN.  TYPE N(O) ',/,
755          1' TO RETURN TO MENU SELECTION AND RESTART THE GAME.')
756           READ(5,200) INPUT
757           CALL INVAT1(INPUT,A)
758           IF(A.EQ.ALPHA(2)) GO TO 1
759           GO TO 19
760     20    WRITE(6,302)
761     302   FORMAT('1RED CALL TO STATUS ROUTINES.  ENTER THE RED PASSWORD.')
762           READ(5,200) INPUT
763           CALL INVAT6(INPUT,CHECK)
764           CALL STATUS(CHECK,ERROR,NAMEWP)
765           IF(ERROR.EQ.0) GO TO 21
766           WRITE(6,301)
767           READ(5,200) INPUT
768           CALL INVAT1(INPUT,A)
769           IF(A.EQ.ALPHA(2)) GO TO 1
770           GO TO 20
771     21    CALL CONTRL
772           BDES=0
773           RDES=0
774           DO 303 I=1,100
775           S=SUNK(I)
776           IF(S.EQ.-1) GO TO 22
777           IF((S.GE.PBSUR1.AND.S.LE.PBSUR2).OR.(S.GE.PBSUB1.AND.S.LE.PBSUB2))
778          1BDES=BDES+1
779           IF((S.GE.PRSUR1.AND.S.LE.PRSUR2).OR.(S.GE.PRSUB1.AND.S.LE.PRSUB2))
780          1RDES=RDES+1
781     303   CONTINUE
782     22    IF(BDES.EQ.(BLUSUR+BLUSUB)) GO TO 9997
783           IF(BDES.LT.BLUDES) GO TO 23
784           WRITE(6,305)
785     305   FORMAT('1THE NUMBER OF BLUE SHIPS DESTROYED IS EQUAL TO OR ',
786          1/,' GREATER THAN THE NUMBER INPUT FOR GAME END.  IF YOU WANT',
787          1/,' TO CONTINUE THE GAME, TYPE Y(ES).  TYPE N(O) TO END THE GAME.'
788          1)
789           READ(5,200) INPUT
790           CALL INVAT1(INPUT,A)
791           IF(A.EQ.ALPHA(2)) GO TO 9997
792     23    IF(RDES.EQ.(REDSUR+REDSUB)) GO TO 9997
793           IF(RDES.LT.REDDES) GO TO 24
794           WRITE(6,306)
795     306   FORMAT('1THE NUMBER OF RED SHIPS DESTROYED IS EQUAL TO OR',
796          1/,' GREATER THAN THE NUMBER INPUT FOR GAME END.  IF YOU WANT',
797          1/,' TO CONTINUE THE GAME, TYPE Y(ES).  TYPE N(O) TO END THE GAME.'
798          1)
799           READ(5,200) INPUT
800           CALL INVAT1(INPUT,A)
```

```
      IF(A.EQ.ALPHA(2)) GO TO 9997
24    ITURN=ITURN+1
      IF(ITURN.LT.TURN) GO TO 25
      WRITE(6,304) ITURN
304   FORMAT('1YOU HAVE JUST COMPLETED GAME TURN ',I3,'.',
     1/,' TYPE Y(ES) IF YOU WANT TO CONTINUE THE GAME, OR N(O) ',
     1/,' TO ALLOW THE GAME TO END.')
      READ(5,200) INPUT
      CALL INVAT1(INPUT,A)
      IF(A.EQ.ALPHA(2)) GO TO 9997
25    GO TO 1
C
C     GAME END
C
9997  WRITE(6,307)
307   FORMAT('1BLUE RESULTS:',//)
      WRITE(6,308)
308   FORMAT(' ',31X,'FINAL',/,5X,'SHIP',5X,'STATUS/HITS',5X,
     1'LOCATION')
      COUNT=0
      IF(BLUSUR.EQ.0) GO TO 27
      A=PBSUR1
      B=PBSUR2
      C=1
26    DO 313 PLT=A,8
      TEMP=-PLATFM(PLT,11)
      IF(PLATFM(PLT,20).EQ.0) WRITE(6,309) (NAME(PLT,J),J=1,6),
     1PLATFM(PLT,3)
309   FORMAT('0',3X,6A1,7X,'SUNK',10X,I4)
      IF(PLATFM(PLT,20).GT.0) WRITE(6,310) (NAME(PLT,J),J=1,6),
     1TEMP,PLATFM((PLT,3)
310   FORMAT('0',3X,6A1,9X,I1,12X,I4)
      COUNT=COUNT+1
      IF(COUNT.LT.10) GO TO 313
      COUNT=0
      WRITE(6,311)
311   FORMAT('0MORE ON NEXT PAGE.  TYPE Y(ES) TO CONTINUE.')
      READ(5,201) G
      WRITE(6,312)
312   FORMAT('1',31X,'FINAL',/,5X,'SHIP',5X,'STATUS/HITS',5X,
     1'LOCATION')
313   CONTINUE
      IF(C.GT.3) GO TO 9998
      IF(C.GT.2) GO TO 29
      IF(C.GT.1) GO TO 28
27    IF(BLUSUB.EQ.0) GO TO 28
      A=PBSUB1
      B=PBSUB2
      C=2
      GO TO 26
```

```
851     28    WRITE(6,314)
852     314   FORMAT('0TYPE Y(ES) TO SEE RED RESULTS.')
853           READ(5,201) G
854           WRITE(6,315)
855     315   FORMAT('1RED RESULTS:',//)
856           WRITE(6,308)
857           COUNT=0
858     316   IF(REDSUB.EQ.0) GO TO 29
859           A=PRSUB1
860           B=PRSUB2
861           C=3
862           GO TO 26
863     29    IF(REDSUB.EQ.0) GO TO 9998
864           A=PRSUB1
865           B=PRSUB2
866           C=4
867           GO TO 26
868     9998  WRITE(6,317)
869     317   FORMAT('0TYPE Y(ES) TO RETURN TO MENU SELECTION AND START A NEW',
870          1/,' GAME.  TYPE N(O) TO EXIT THE GAME.')
871           READ(5,200) INPUT
872           CALL INVRT1(INPUT,A)
873           IF(A.EQ.ALPHA(1)) GO TO 1
874     9999  WRITE(6,318)
875     318   FORMAT('0EXITING THE GAME.')
876           STOP
877           END
```

```
1            SUBROUTINE STATUS(PASWRD,ERROR,NAMEWP)
2       C
3       C    THIS SUBROUTINE IS THE DRIVER FOR ALL DISPLAY ROUTINES.
4       C
5            IMPLICIT INTEGER (A-Z)
6            DIMENSION PASWRD(6),INPUT(70),ALPHA(2),NAMEWP(29,6)
7            COMMON FLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
8           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
9           2IPLAT,IHIR,ISURF,ISUB,AOV,POV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
10          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
11          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
12           DATA ALPHA/'Y','N'/
13           A=0
14           ERROR=0
15           DO 100 I=1,6
16           IF(PASWRD(I).NE.BLUPAS(I)) A=1
17           IF(A.EQ.1) GO TO 200
18      100  CONTINUE
19      200  IF(A.EQ.1) GO TO 300
20           AC1=1
21           AC2=BLUAIR
22           SUR1=PBSUR1
23           SUR2=PBSUR2
24           SUB1=PBSUB1
25           SUB2=PBSUB2
26           AC=BLUAIR
27           SUR=BLUSUR
28           SUB=BLUSUB
29           SIDE=1
30           GO TO 800
31      300  A=0
32           DO 400 I=1,6
33           IF(PASWRD(I).NE.REDPAS(I)) A=1
34           IF(A.EQ.1) GO TO 500
35      400  CONTINUE
36      500  IF(A.EQ.1) GO TO 600
37           AC1=PRAIR1
38           AC2=PRAIR2
39           SUR1=PRSUR1
40           SUR2=PRSUR2
41           SUB1=PRSUB1
42           SUB2=PRSUB2
43           AC=REDAIR
44           SUR=REDSUR
45           SUB=REDSUB
46           SIDE=2
47           GO TO 800
48      600  ERROR=1
49           WRITE(6,700)
50      700  FORMAT('0YOU HAVE MADE AN ERROR IN THE ENTRY OF YOUR PASSWORD.')
```

```
      GO TO 9999
800   WRITE(6,821)
821   FORMAT('1THE FOLLOWING DISPLAY OPTIONS ARE AVAILABLE TO YOU:',
     1///,' 1. DETECTIONS OF ENEMY PLATFORMS.',
     1//,' 2. RESULTS OF WEAPONS FIRED.',
     1//,' 3. DAMAGE TO FRIENDLY PLATFORMS.',
     1//,' 4. STATUS OF FRIENDLY PLATFORMS AND INPUT OF INSTRUCTIONS.',
     1//,' 5. BEGIN NEXT GAME TURN.')
1000  WRITE(6,824)
824   FORMAT(///,' SELECT AND ENTER THE NUMBER OF A DISPLAY OPTION.',
     1/,' ENTER A 6 FOR A LIST OF DISPLAY OPTIONS.')
      READ 5,822) INPUT
822   FORMAT(70A1)
      CALL NONVAT(INPUT,A)
      IF(A.EQ.6) GO TO 800
      IF(A.EQ.1.OR.A.EQ.2.OR.A.EQ.3.OR.A.EQ.4.OR.A.EQ.5) GO TO 830
      WRITE(6,823)
823   FORMAT('0YOU HAVE MADE AN ERROR IN YOUR INPUT SELECTION.  TRY AGAI
     1N.')
      GO TO 1000
830   IF(A.GT.1) GO TO 2
      CALL DETXT(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP)
      GO TO 1000
2     IF(A.GT.2) GO TO 3
      CALL SHOTRE(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP,AC,SUR,SUB)
      GO TO 1000
3     IF(A.GT.3) GO TO 4
      CALL DAMRE(SUR1,SUR2,SUB1,SUB2,SUR,SUB)
      GO TO 1000
4     IF(A.GT.4) GO TO 5
      CALL STATIN(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP)
      GO TO 1000
5     WRITE(6,831)
831   FORMAT('0THIS OPTION WILL BEGIN THE NEXT GAME TURN.  TYPE Y(ES)',
     1/,' IF YOU HAVE ENTERED ALL YOUR COMBAT, MOVE AND SEARCH ',
     1/,' INSTRUCTIONS AND YOU ARE READY TO COMMENCE PLAY.  TYPE N(O)',
     1/,' TO RETURN TO DISPLAY SELECTION.')
      READ(5,822) INPUT
      CALL INVAT1(INPUT,B)
      IF(B.NE.ALPHA(1)) GO TO 1000
9999  RETURN
      END
```

```
      SUBROUTINE OETXT(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP)
C
C     THIS SUBROUTINE DISPLAYS THE OETECTIONS OF THE CALLING FORCE.
C
      IMPLICIT INTEGER (A-Z)
      COMMON PLATFM(100,37),OETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
     1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REOPAS(6),
     2IPLAT,IAIR,ISURF,ISUB,ADV,POV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
     3REDSUR,BLUSUB,REOSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
     4PRSUR),PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
      DIMENSION ALPHA1(2,5),ALPHA2(2,4),NAMEWP(29,6)
      DATA ALPHA1/'SU','SU','RF','BM','AC','AR','E ','IN','  ','E '/
      DATA ALPHA2/'PA','AC','SS','TI','IV','VE','E ',' '/
      COUNT=0
      WRITE(6,1)
1     FORMAT('1',14X,'SONOR',7X,'SHIP',/,2X,'OETECTING',4X,'TYPE',8X,
     1'TYPE',/,2X,'PLATFORM',5X,'USED',6X,'OETECTED',3X,'LOCATION',/)
      DO 6 PLT=1,100
      D1=OETECT(PLT,1)
      D2=OETECT(PLT,2)
      IF(D1.EQ.-1) GO TO 9999
      IF((D1.GE.AC1.AND.D1.LE.AC2).OR.(D1.GE.SUR1.AND.D1.LE.SUR2).OR.
     1(D1.GE.SUB1.AND.D1.LE.SUB2)) GO TO 2
      GO TO 6
2     WRITE(6,3)  (NAME(D1,J),J=1,6),(ALPHA2(OETECT(PLT,3)+1,J),J=1,4),
     1(ALPHA1(PLATFM(D2,8),J),J=1,5),PLATFM(D2,3)
3     FORMAT(' ',3X,6A1,5X,4A2,3X,5A2,3X,I4)
      COUNT=COUNT+1
      IF(COUNT.LT.19) GO TO 6
      COUNT=0
      WRITE(6,4)
4     FORMAT('0MORE OETECTIONS ON NEXT PAGE.  TYPE Y(ES) TO CONTINUE.')
      READ(5,5) C
5     FORMAT(A1)
      WRITE(6,1)
6     CONTINUE
9999  WRITE(6,7)
7     FORMAT('0ENO OF DETECTION ROUTINE.  TYPE Y(ES) TO CONTINUE.')
      READ(5,5) C
      RETURN
      ENO
```

```
1            SUBROUTINE SHOTRE(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP,AC,SUR,SUB)
2       C
3       C    THIS SUBROUTINE DISPLAYS THE RESULTS OF WEAPONS FIRED BY THE
4       C    CALLING FORCE.
5       C
6            IMPLICIT INTEGER (A-Z)
7            REAL R
8            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
9           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
10          2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
11          3REDSUR BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
12          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
13           DIMENSION ALPHA1(2,7),NAMEWP(29,6)
14           DATA ALPHA1/'DE','NO','TO',' D','NA','ET','TI','ON','ON','AT','  '
15          1,'IO','  ','N '/
16           IF(AC.EQ.0) GO TO 20
17           WRITE(6,1)
18      1    FORMAT('1')
19           COUNT=0
20           DO 10 PLT=AC1,AC2
21           IF((PLATFM(PLT,23).EQ.0).AND.(PLATFM(PLT,24).EQ.0)) GO TO 10
22           WRITE(6,4)  (NAME(PLT,J),J=1,6),(NAME(PLATFM(PLT,21),J),J=1,6),PLAT
23          1FM(PLT,2)
24      4    FORMAT(//,' AIRCRAFT - ',6A1,5X,'PARENT SHIP - ',6A1,5X,
25          1'COMBAT LOCATION - ',I4)
26           DO 7 WP=1,2
27           RESULT=PLATFM(PLT,WP+22)
28           IF(RESULT.EQ.0) GO TO 7
29           R=RESULT/6.
30           IF(RESULT.GE.1.AND.RESULT.LE.6) WRITE(6,5) WP,(NAMEWP(PLATFM(PLT,W
31          1P+16),J),J=1,6),R
32      5    FORMAT(' WEAPON ',I1,3X,'TYPE - ',6A2,5X,'PROB(HIT)/DETONATION - '
33          1,F6.4)
34           IF(RESULT.GT.6) WRITE(6,6) WP,(NAMEWP(PLATFM(PLT,WP+16),J),J=1,6),
35          1(ALPHA1((RESULT-6),J),J=1,7)
36      6    FORMAT(' WEAPON ',I1,3X,'TYPE - ',6A2,5X,'PROB(HIT)/DETONATION - '
37          1,7A2)
38      7    CONTINUE
39           COUNT=COUNT+1
40           IF(COUNT.LT.5) GO TO 10
41           WRITE(6,8)
42      8    FORMAT('0MORE ON NEXT PAGE.  TYPE Y(ES) TO CONTINUE.')
43           READ(5,9) G
44      9    FORMAT(A1)
45           COUNT=0
46      10   CONTINUE
47      11   WRITE(6,12)
48      12   FORMAT(//,' TYPE Y(ES) FOR SHIP RESULTS.')
49           READ(5,9) G
50      20   WRITE(6,1)
```

```
      COUNT=0
      IF(SUA.EQ.0) GO TO 18
      A=SUA1
      B=SUA2
      C=1
13    DO 16 PLT=A,B
      IF((PLATFM(PLT,35).EQ.0).AND.(PLATFM(PLT,36).EQ.0).AND.
     1(PLATFM(PLT,37).EQ.0)) GO TO 16
      WRITE(6,14) (NAME(PLT,J),J=1,6),PLATFM(PLT,2)
14    FORMAT(//,' SHIP - ',6A1,5X,'COMBAT LOCATION - ',I4)
      DO 15 WP=1,3
      RESULT=PLATFM(PLT,WP+34)
      IF(RESULT.EQ.0) GO TO 15
      R=RESULT/6.
      IF(RESULT.GE.1.AND.RESULT.LE.6) WRITE(6,5) WP,
     1(NAMEWP(PLATFM(PLT,WP+20),J),J=1,6),R
      IF(RESULT.GT.6) WRITE(6,6) WP,(NAMEWP(PLATFM(PLT,WP+20),J),J=1,6),
     1(ALPHA1((RESULT-6),J),J=1,7)
15    CONTINUE
      COUNT=COUNT+1
      IF(COUNT.LT.4) GO TO 16
      WRITE(6,8)
      READ(5,9) G
      COUNT=0
16    CONTINUE
      IF(C.GT.1) GO TO 9999
18    IF(SUB.EQ.0) GO TO 9999
      C=2
      A=SUB1
      B=SUB2
      GO TO 13
9999  WRITE(6,17)
17    FORMAT('0END OF WEAPON RESULT DISPLAY.  TYPE Y(ES) TO CONTINUE.')
      READ(5,9) G
      RETURN
      END
```

```
1             SUBROUTINE DAMRE(SUR1,SUR2,SUB1,SUB2,SUR,SUB)
2       C
3       C     THIS SUBROUTINE DISPLAYS DAMAGE SUFFERED BY THE CALLING FORCE
4       C
5             IMPLICIT INTEGER (A-Z)
6             COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
7            1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
8            2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
9            3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
10           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
11            WRITE(6,1)
12      1     FORMAT('1',5X,'SHIP',5X,'STATUS/HITS')
13            COUNT=0
14            IF(SUR.EQ.0) GO TO 9
15            A=SUR1
16            B=SUR2
17            C=1
18      2     DO 7 PLT=A,B
19            IF(PLATFM(PLT,20).EQ.0) WRITE(6,3)  (NAME(PLT,J),J=1,6)
20      3     FORMAT('0',4X,6A1,7X,'SUNK')
21            TEMP=-PLATFM(PLT,11)
22            IF(PLATFM(PLT,20).GT.0) WRITE(6,4)  (NAME(PLT,J),J=1,6),TEMP
23      4     FORMAT('0',4X,6A1,9X,I1)
24            COUNT=COUNT+1
25            IF(COUNT.LT.10) GO TO 7
26            COUNT=0
27            WRITE(6,5)
28      5     FORMAT('0MORE ON NEXT PAGE.  TYPE Y(ES) TO CONTINUE.')
29            READ(5,6) G
30      6     FORMAT(A1)
31            WRITE(6,1)
32      7     CONTINUE
33            IF(C.GT.1) GO TO 9999
34      9     IF(SUB.EQ.0) GO TO 9999
35            C=2
36            A=SUB1
37            B=SUB2
38            GO TO 2
39      9999  WRITE(6,8)
40      8     FORMAT('0END OF DAMAGE DISPLAY ROUTINE.  TYPE Y(ES) TO CONTINUE.')
41            READ(5,6) G
42            RETURN
43            END
```

```
      SUBROUTINE STATIN(AC1,AC2,SUR1,SUR2,SUB1,SUB2,NAMEWP)
C
C
C        DISPLAYS INDIVIDUAL PLATFORM STATUS OF COMBAT ACTIONS, MOVEMENT
C             AND SEARCH RESULTS.  ACCEPTS COMBAT, MOVEMENT AND SEARCH
C             COMMANDS. ALL INPUT IS PUT INTO A BUFFER AND OVERLAYED AT
C             GAME TURN EXECUTION
C
C
      IMPLICIT INTEGER(A-Z)
      COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
     1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
     2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
     3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
     4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
      DIMENSION INPUT(70),INAME(6),NAMEWP(29,6),EXIT(6)
      DATA NO/'N'/
5     WRITE(6,100)
100   FORMAT(//' YOU HAVE SELECTED INDIVIDUAL PLATFORM STATUS DISPLAY',
     1' AND ASSOCIATED'/' COMBAT,MOVEMENT AND SEARCH COMMAND INPUTS.'/
     2' ENTER THE NAME OF THE DESIRED PLATFORM.'//)
10    READ(5,101) INPUT
101   FORMAT(70A1)
      CALL INVRT6(INPUT,INAME)
      CALL NAMEIN(INAME,INDEX,NAME,AC1,AC2,SUR1,SUR2,SUB1,SUB2)
      IF(INDEX.NE.0)GO TO 20
      WRITE(6,102)
102   FORMAT(' YOU HAVE ENTERED AN INCORRECT PLATFORM NAME.'/T5,'REENTER
     1'///)
      GO TO 10
20    TEST=0
      IF(INDEX.GE.AC1.AND.INDEX.LE.AC2)CALL STATAC(TEST,INDEX,INAME,
     1NAMEWP)
      IF((INDEX.GE.SUR1.AND.INDEX.LE.SUR2).OR.(INDEX.GE.SUB1.AND.INDEX.
     1LE.SUB2))CALL STATSS(TEST,INDEX,INAME,NAMEWP)
      IF(TEST.EQ.0)WRITE(6,103)
103   FORMAT(' YOU CAN ONLY OBTAIN STATUS AND INPUT COMMANDS'/
     1' FOR PLATFORMS UNDER YOUR CONTROL.')
      WRITE(6,104)
104   FORMAT(//' ENTER Y(ES) TO CONTINUE WITH ANOTHER PLATFORM'/
     1' ENTER N(O) TO RETURN TO DISPLAY SELECTION')
      READ(6,101) INPUT
      CALL INVRT1(INPUT,ATEST)
      IF(ATEST.EQ.NO)RETURN
      WRITE(6,106)
106   FORMAT(//' ENTER THE NAME OF THE DESIRED PLATFORM'//)
      GO TO 10
      END
```

```
      SUBROUTINE STATAC(USED,INOEX,INAME,NAMEWP)
C
C
C        AIRCRAFT STATUS AND COMMAND INPUT SUBROUTINE
C
C
      IMPLICIT INTEGER(A-Z)
      COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
     1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
     2IPLAT,IAIR,ISURF,ISUB,AOV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
     3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
     4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
      DIMENSION INAME(6),NAMEWP(29,6),INPUT(70),WEPON(6)
      DATA NONE/'N'/
      USED=1
      IF(PLATFM(INOEX,11).EQ.0)WRITE(6,100)INAME
  100 FORMAT(//1X,6A1,' IS CURRENTLY NOT FLYING - IT IS ASSUMED YOU'/
     1' DESIRE TO LAUNCH.')
      IF(PLATFM(INDEX,11).EQ.0)GO TO 35
C
C
C        STATUS DISPLAYS
C
C
      WRITE(6,101)INAME,PLATFM(INOEX,3),PLATFM(INOEX,4)
  101 FORMAT(//1X,6A1,' IS LOCATED AT POSITION',I5,' AT SPEED',I3)
      IF(PLATFM(INDEX,23).EQ.0.AND.PLATFM(INDEX,24).EQ.0)GO TO 10
      X=23
      IF(PLATFM(INDEX,23).EQ.0)X=24
      WRITE(6,102)(NAMEWP(PLATFM(INOEX,X-6),J),J=1,6),PLATFM(INDEX,2)
  102 FORMAT(' IT RELEASED A ',6A2,' AT A SUBMARINE TARGET LOCATED'/
     1' AT POSITION',I5/' THE WEAPON')
      IF(PLATFM(INDEX,X).LE.6)WRITE(6,103)PLATFM(INOEX,X)
  103 FORMAT(' DETONATED WITH PROBABILITY OF A HIT OF',I2,
     1' OUT OF 6')
      IF(PLATFM(INOEX,X).EQ.7)WRITE(6,104)
  104 FORMAT(T13,'DETONATED  (HIT) ')
      IF(PLATFM(INOEX,X).EQ.8)WRITE(6,105)
  105 FORMAT(T13,'DID NOT DETONATE')
   10 IF(PLATFM(INOEX,7).EQ.0)WRITE(6,106)
  106 FORMAT(' IT WAS NOT ASSIGNED A SEARCH MISSION')
      IF(PLATFM(INOEX,7).EQ.0)GO TO 30
      IF(PLATFM(INDEX,4).GT.0)WRITE(6,107)INAME
  107 FORMAT(1X,6A1,' COULD NOT EXECUTE A SEARCH MISSION AS ITS SPEED'/
     1' WAS GREATER THAN ZERO')
      IF(PLATFM(INDEX,4).GT.0)GO TO 30
      IF(PLATFM(INOEX,7).EQ.1)WRITE(6,108)
  108 FORMAT(' IT CONDUCTED AN ACTIVE SEARCH WITH FOLLOWING RESULTS'/
     1' SUBMARINE CONTACT LOCATION    RANGE')
      IF(PLATFM(INDEX,7).EQ.2)WRITE(6,109)
```

```
  109   FORMAT(' IT CONDUCTED A PASSIVE SEARCH WITH FOLLOWING RESULTS'/
     1' SUBMARINE CONTACT LOCATION    RANGE')
        X=0
        DO 20 I=1,100
        IF(DETECT(I,1).LT.0)GO TO 21
        IF(DETECT(I,1).NE.INDEX)GO TO 20
        X=1
        WRITE(6,110)PLATFM(DETECT(I,2),3),DETECT(I,4)
  110   FORMAT(T13,I4,T31,I4)
  20    CONTINUE
  21    IF(X.EQ.0)WRITE(6,111)
  111   FORMAT(T5,'THERE WERE NO DETECTIONS')
        WRITE(6,127)(NAME(INDEX,J),J=1,6)
  127   FORMAT(1X,6A1,' WEAPON SYSTEM(S) '/T5,'TYPE',T15,'RANGE')
        WRITE(6,128)(NAMEWP(PLATFM(INDEX,17),J),J=1,6),WEP(PLATFM(INDEX,
     117),2)
  128   FORMAT(' 1   ',6A2,I3)
        IF(PLATFM(INDEX,18).EQ.0)GO TO 30
        WRITE(6,129)(NAMEWP(PLATFM(INDEX,18),J),J=1,6),WEP(PLATFM(INDEX,
     118),2)
  129   FORMAT(' 2   ',6A2,I3)
C
C
C         COMMAND INPUTS
C
C
  30    WRITE(6,112)
  112   FORMAT(//' NO COMBAT ENTER N(ONE)     OTHERWISE ENTER'/
     1' TARGET LOCATION'/' WEAPON SYSTEM NUMBER      IN THIS FASHION'//)
C
C
C
C               COMBAT
C
C
        READ(5,113)INPUT
  113   FORMAT(70A1)
        CALL INVRT1(INPUT,TEST)
        IF(TEST.EQ.NONE)GO TO 40
        CALL NONVRT(INPUT,BUFFER(INDEX,2))
        READ(5,113)INPUT
        CALL NONVRT(INPUT,BUFFER(INDEX,1))
        GO TO 40
C
C
C               LAUNCH AIRCRAFT
C
C
  35    WRITE(6,114)INAME
  114   FORMAT(//1X,6A1,' IS NOW BEING LAUNCHED'//)
```

```
101           PLATFM(INDEX,11)=1
102     C
103     C
104     C                MOVEMENT
105     C
106     C
107     40    WRITE(6,115)
108     115   FORMAT(' MOVEMENT NOT DESIRED ENTER N(ONE)   OTHERWISE ENTER'/
109          1' SPEED'/' DESTINATION HEX          IN THIS FASHION'//)
110           READ(5,113) INPUT
111           CALL INVRT1(INPUT,TEST)
112           IF(TEST.EQ.NONE)BUFFER(1,4)=PLATFM(1,3)
113           IF(TEST.EQ.NONE)GO TO 50
114           CALL NONVRT(INPUT,BUFFER(INDEX,3))
115           READ(5,113) INPUT
116           CALL NONVRT(INPUT,BUFFER(INDEX,4))
117     C
118     C
119     C                TEST FOR GOING RIGHT DISTANCE AND NOT EXCEEDING MAX SPEED
120     C
121     C
122           CALL DISTAN(BUFFER(INDEX,4),PLATFM(INDEX,3),DIST)
123           IF(DIST.LE.BUFFER(INDEX,3).AND.BUFFER(INDEX,3).LE.PLATFM(INDEX,12)
124          1)GO TO 50
125           WRITE(6,116)
126     116   FORMAT(' YOU HAVE EITHER SET A SPEED HIGHER THAN MAX'/'  OR ATTEMP
127          1TED TO GO FURTHER THAN SET SPEED')
128           GO TO 40
129     C
130     C
131     C                SEARCH
132     C
133     C
134     50    WRITE(6,117)
135     117   FORMAT(/' ENTER SEARCH MISSIONS AS FOLLOWS:'/T7,'0  NONE SCHEDULED
136          1'/T7,'1  ACTIVE MISSION'/T7,'2  PASSIVE MISSION'//)
137           READ(5,113) INPUT
138           CALL NONVRT(INPUT,X)
139           IF(X.GT.0.AND.BUFFER(INDEX,3).GT.0)WRITE(6,118)
140     118   FORMAT(' A SEARCH MISSION CANNOT BE PERFORMED WHILE THIS PLATFORM'
141          1/' IS MOVING (NOT HOVERING OR LOITERING).'/)
142           IF(X.GT.0.AND.BUFFER(INDEX,3).GT.0)GO TO 50
143           BUFFER(INDEX,5)=X
144     C
145     C
146     C        COMMAND STATUS/INPUT COMPLETION
147     C
148     C
149           WRITE(6,119)
150     119   FORMAT(' YOU HAVE ENTERED THE FOLLOWING COMMANDS'/' COMBAT:')
```

```
151           IF(BUFFER(INDEX,1).EQ.0)WRITE(6,120)
152     120   FORMAT(' NONE SCHEDULED')
153           IF(BUFFER(INDEX,1).GT.0)WRITE(6,121)(NAMEWP(PLATFM(INDEX,BUFFER(IN
154          1DEX,1)+16),J),J=1,6),BUFFER(INDEX,2)
155     121   FORMAT(1X,6A2,' WEAPON     TGT LOCATION',I5)
156           IF(BUFFER(INDEX,3).EQ.0)WRITE(6,122)PLATFM(INDEX,3)
157     122   FORMAT(' SPEED IS ZERO AT LOCATION ',I4)
158           IF(BUFFER(INDEX,3).GT.0)WRITE(6,123)BUFFER(INDEX,4),BUFFER(INDEX,
159          13)
160     123   FORMAT(' DESTINATION',I5,' SPEED',I3)
161           IF(BUFFER(INDEX,5).EQ.0)WRITE(6,124)
162     124   FORMAT(' THERE IS NO SEARCH MISSION SCHEDULED'/)
163           IF(BUFFER(INDEX,5).EQ.1)WRITE(6,125)
164     125   FORMAT(' AN ACTIVE SEARCH MISSION WILL BE CONDUCTED'/)
165           IF(BUFFER(INDEX,5).EQ.2)WRITE(6,126)
166     126   FORMAT(' A PASSIVE SEARCH MISSION WILL BE CONDUCTED'/)
167           RETURN
168           END
```

```
1            SUBROUTINE STATSS(USED,INDEX,INAME,NAMEWP)
2       C
3       C
4       C        DISPLAYS INDIVIDUAL SUFACE SHIP/SUBMARINE STATUS OF COMBAT
5       C             ACTIONS, MOVEMENT AND SEARCH RESULTS.  ACCEPTS COMBAT
6       C             MOVEMENT AND SEARCH COMMANDS.  ALL INPUT IS PLACED IN
7       C             A BUFFER AND OVERLAYED AT GAME TURN EXECUTION
8       C
9       C
10           IMPLICIT INTEGER(A-Z)
11           COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
12          1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
13          2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
14          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
15          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
16           DIMENSION INAME(6),NAMEWP(29,6),INPUT(70)
17           DATA NONE/'N'/
18           USED=1
19      C
20      C
21      C        STATUS DISPLAYS
22      C
23      C
24           WRITE(6,100)INAME,PLATFM(INDEX,3),PLATFM(INDEX,4)
25      100  FORMAT(///1X,6A1,' IS LOCATED AT POSITION',I5,' SPEED',I2)
26           TEMP=-PLATFM(INDEX,11)
27           IF(PLATFM(INDEX,11).LT.0)WRITE(6,99)PLATFM(INDEX,12),TEMP
28      99   FORMAT(//T10,'NOTE'/' DUE TO PREVIOUS AND/OR CURRENT DAMAGE SUFFER
29          1ED, MAX SPEED IS NOW',I2/' THE SHIP HAS BEEN HIT',I2,' TIMES'//)
30      C
31      C
32      C        PLATFORM HIT LAST GAME TURN?
33      C
34      C
35           DO 10 I=1,100
36           IF(HIT I,1).LT.0)GO TO 11
37           IF(HIT(I,1).NE.INDEX)GO TO 10
38           IF(WEP(HIT(I,2),3).NE.3)WRITE(6,101)
39      101  FORMAT(' IT HAS BEEN HIT BY A NON-TORPEDO WEAPON')
40           IF(WEP(HIT(I,2),3).EQ.3)WRITE(6,102)
41      102  FORMAT(' IT HAS BEEN HIT BY A TORPEDO')
42      10   CONTINUE
43      C
44      C
45      C        COMBAT ACTION RESULTS
46      C
47      C
48      11   IF(PLATFM(INDEX,1).EQ.0)GO TO 30
49           WRITE(6,103)INAME,PLATFM(INDEX,2)
50      103  FORMAT(/T5,'THE ',6A1,' ATTACK ON TARGET LOCATED AT',I5/)
```

```
      DO 20 I=35,37
      IF(PLATFM(INDEX,I).EQ.0)GO TO 20
      WRITE(6,104) (NAMEWP(PLATFM(INDEX,I-14),J),J=1,6)
104   FORMAT(' RELEASE OF ',6A2,' RESULTED IN')
      IF(PLATFM(INDEX,I).LE.6)WRITE(6,105)PLATFM(INDEX,I)
105   FORMAT(' A DETONATION WITH PROBABILITY'/' OF A HIT OF ',
     1I1,' OUT OF 6'/)
      IF(PLATFM(INDEX,I).EQ.7)WRITE(6,106)
106   FORMAT(' A DETONATION(HIT) '/)
      IF(PLATFM(INDEX,I).EQ.8)WRITE(6,107)
107   FORMAT(' NO DETONATION - NO DISTURBED WATER MARKER'/)
20    CONTINUE
C
C
C        SEARCH MISSION STATUS
C
C
30    IF(PLATFM(INDEX,7).EQ.0)WRITE(6,108)
108   FORMAT(' IT WAS NOT ASSIGNED A SEARCH MISSION')
      IF(PLATFM(INDEX,7).EQ.0)GO TO 50
      WRITE(6,109)
109   FORMAT(' IT WAS ASSIGNED THE FOLLOWING SEARCH MISSION(S) ')
      IF(PLATFM(INDEX,7).EQ.1)WRITE (6,110)
110   FORMAT(' ACTIVE')
      IF(PLATFM(INDEX,7).EQ.2)WRITE (6,111)
111   FORMAT(' PASSIVE')
      IF(PLATFM(INDEX,7).EQ.3)WRITE(6,112)
112   FORMAT(' ACTIVE AND PASSIVE')
      WRITE(6,113)
113   FORMAT('    TYPE    LOCATION   METHOD    RANGE')
      X=0
      DO 40 I=1,100
      IF(DETECT(I,1).LT.0)GO TO 41
      IF(DETECT(I,1).NE.INDEX)GO TO 40
      X=1
      IF(PLATFM(DETECT(I,2),8).EQ.1.AND.DETECT(I,3).EQ.0)WRITE(6,1141)
     1PLATFM(DETECT(I,2),3),DETECT(I,4)
1141  FORMAT(' SURFACE',5X,I4,5X,'PASSIVE',I6)
      IF(PLATFM(DETECT(I,2),8).EQ.2.AND.DETECT(I,3).EQ.0)WRITE(6,1142)
     1PLATFM(DETECT(I,2),3),DETECT(I,4)
1142  FORMAT(' SUBMARINE',3X,I4,5X,'PASSIVE',I6)
      IF(PLATFM(DETECT(I,2),8).EQ.1.AND.DETECT(I,3).EQ.1)WRITE(6,1151)
     1PLATFM(DETECT(I,2),3),DETECT(I,4)
1151  FORMAT(' SURFACE',5X,I4,5X,'ACTIVE',I7)
      IF(PLATFM(DETECT(I,2),8).EQ.2.AND.DETECT(I,3).EQ.1)WRITE(6,1152)
     1PLATFM(DETECT(I,2),3),DETECT(I,4)
1152  FORMAT(' SUBMARINE',3X,I4,5X,'ACTIVE',I7)
40    CONTINUE
41    IF(X.EQ.0)WRITE(6,116)
116   FORMAT(' THERE WERE NO DETECTIONS')
```

```
101     C
102     C
103     C         IF SUNK - SKIP COMMAND INPUTS
104     C
105     C
106     50     IF(PLATFM(INDEX,20).EQ.0)WRITE(6,117)INAME
107     117    FORMAT(//' SINCE THE ',6A1,' HAS BEEN SUNK, NO COMMAND INPUTS',
108           1' CAN BE ACCEPTED')
109            IF(PLATFM(INDEX,20).EQ.0)RETURN
110            WRITE(6,134)(NAME(INDEX,J),J=1,6)
111     134    FORMAT(1X,6A1,' WEAPON SYSTEM(S) '/T5,'TYPE',T15,'RANGE')
112            DO 51 I=1,3
113            II=I+20
114            IF(PLATFM(INDEX,II).EQ.0)GO TO  52
115            WRITE(6,135)I,(NAMEWP(PLATFM(INDEX,II),J),J=1,6),WEP(PLATFM(INDEX,
116           1II),2)
117     135    FORMAT(I2,2X,6A2,I3)
118     51     CONTINUE
119     C
120     C
121     C         COMMAND INPUTS
122     C
123     C
124     52     WRITE(6,118)
125     118    FORMAT(//' NO COMBAT ENTER N(ONE) - OTHERWISE ENTER'/' TARGET LOCA
126           1TION'/' WEAPON SYSTEM NUMBER   IN THIS FASHION'//)
127            READ(5,119)INPUT
128     119    FORMAT(70A1)
129            CALL INVRT1(INPUT,TEST)
130            IF(TEST.EQ.NONE)GO TO 70
131            CALL NONVRT(INPUT,BUFFER(INDEX,2))
132            READ(5,119)INPUT
133     60     CALL NONVRT(INPUT,WEPON)
134            BUFFER(INDEX,1)=1
135            BUFFER(INDEX,WEPON+5)=-1
136            WRITE(6,120)
137     120    FORMAT(' TO EMPLOY ANOTHER WEAPON SYSTEM, ENTER THE SYSTEM NUMBER'
138           1/' OTHERWISE ENTER N(O) '//)
139            READ(5,119)INPUT
140            CALL INVRT1(INPUT,TEST)
141            IF(TEST.EQ.NONE)GO TO 70
142            GO TO 60
143     C
144     C
145     C         MOVEMENT COMMANDS
146     C
147     C
148     70     WRITE(6,121)
149     121    FORMAT(//' TO SCHEDULE MOVEMENT ENTER'/' SPEED'/' DESTINATION
150           1  IN THAT ORDER, OTHERWISE ENTER N(ONE)'//)
```

```
      READ(5,119) INPUT
      CALL  INVRT1 (INPUT,TEST)
      IF(TEST.EQ.NONE) BUFFER(INDEX,4)=PLATFM(INDEX,3)
      IF(TEST.EQ.NONE) GO TO 80
      CALL  NONVRT(INPUT,BUFFER(INDEX,3))
      READ(5,119) INPUT
      CALL  NONVRT(INPUT,BUFFER(INDEX,4))
C
C
C        TEST FOR GOING RIGHT DISTANCE AND NOT EXCEEDING MAX SPEED
C
C
      CALL DISTAN(BUFFER(INDEX,4),PLATFM(INDEX,3),DIST)
      IF(DIST.LE.BUFFER(INDEX,3).AND.BUFFER(INDEX,3).LE.PLATFM(INDEX,12)
     1) GO TO 80
      WRITE(6,122)
122   FORMAT(' YOU HAVE EITHER SET A SPEED HIGHER THAN MAX CAPABLE OR AT
     1TEMPTED'/' TO GO FURTHER THAN THE SPEED SETTING WILL ALLOW')
      GO TO 70
C
C
C        SEARCH COMMANDS
C
C
80    WRITE(6,123)
123   FORMAT(//' ENTER SEARCH MISSIONS AS FOLLOWS:'/T7,'0  NONE SCHEDULE
     1D'/T7,'1  ACTIVE MISSION'/T7,'2  PASSIVE MISSION'/T7,'3  ACTIVE AN
     2D PASSIVE'//)
      READ(5,119) INPUT
      CALL  NONVRT(INPUT,X)
      IF(X.EQ.3.AND.PLATFM(INDEX,24).NE.1) WRITE(6,124)
124   FORMAT(' AN ACTIVE AND PASSIVE SEARCH MISSION CANNOT BE PERFORMED
     1 SIMULTANEOUSLY'/' BY THIS PLATFORM')
      IF(X.EQ.3.AND.PLATFM(INDEX,24).NE.1) GO TO 80
      BUFFER(INDEX,5)=X
C
C
C        STATUS/COMMAND INPUT COMPLETION
C
C
      WRITE(6,125)
125   FORMAT(/' YOU HAVE ENTERED THE FOLLOWING COMMANDS'/' COMBAT:')
      IF(BUFFER(INDEX,1).EQ.0) WRITE(6,126)
126   FORMAT(T10,'NONE SCHEDULED')
      IF(BUFFER(INDEX,1).EQ.0) GO TO 91
      DO 90 I=6,8
      IF(BUFFER(INDEX,I).GE.1) GO TO 90
      WRITE(6,127) (NAMEWP(PLATFM(INDEX,I+15),J),J=1,6),BUFFER(INDEX,2)
127   FORMAT(T10,6A2,' WEAPON  TGT LOCATION',I5)
90    CONTINUE
```

```
201     91     IF(BUFFER(INDEX,3).EQ.0)WRITE(6,128)PLATFM(INDEX,3)
202     128    FORMAT(' SPEED ZERO   LOCATION',I5)
203            IF(BUFFER(INDEX,3).GT.0)WRITE(6,129)BUFFER(INDEX,4),BUFFER(INDEX,3
204           1)
205     129    FORMAT(' DESTINATION ',I4,'     SPEED',I2)
206            X=BUFFER(INDEX,5)
207            IF(X.EQ.0)WRITE(6,130)
208     130    FORMAT(' NO SEARCH MISSION')
209            IF(X.EQ.1)WRITE(6,131)
210     131    FORMAT(' ACTIVE SEARCH MISSION')
211            IF(X.EQ.2)WRITE(6,132)
212     132    FORMAT(' PASSIVE SEARCH MISSION')
213            IF(X.EQ.3)WRITE(6,133)
214     133    FORMAT(' ACTIVE AND PASSIVE SEARCH MISSION')
215            RETURN
216            END
```

```
1            SUBROUTINE CONTRL
2       C
3       C
4       C     THIS SUBROUTINE EXECUTES A GAME TURN OF COMBAT,MOVE AND SEARCH
5       C
6       C
7            IMPLICIT INTEGER (A-H,P-Z)
8            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
9           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
10          2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
11          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
12          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
13      C
14      C
15      C        WORKING MATRIX INITIALIZATION
16      C
17      C
18           DO 10 I=1,100
19           WATER(I)=-1
20           HIT(I,1)=-1
21      10   CONTINUE
22           DO 20 I=1,IPLAT
23           PLATFM(I,1)=0
24           PLATFM(I,6)=0
25      20   CONTINUE
26           IF(IAIR.EQ.0)GO TO 21
27           DO 21 I=1,IAIR
28           PLATFM(I,23)=0
29           PLATFM(I,24)=0
30      21   CONTINUE
31           J=IAIR+1
32           DO 22 I=J,IPLAT
33           PLATFM(I,35)=0
34           PLATFM(I,36)=0
35           PLATFM(I,37)=0
36      22   CONTINUE
37      C
38      C
39      C        MOVES NONFLYING AIRCRAFT WITH MOTHER SHIP
40      C
41      C
42           IF(IAIR.EQ.0)GO TO 24
43           DO 24 I=1,IAIR
44           IF(PLATFM(I,11).EQ.1)GO TO 24
45           BUFFER(I,4)=BUFFER(PLATFM(I,21),4)
46      24   CONTINUE
47      C
48      C
49      C             OVERLAY BUFFER INTO PLATFM
50      C
```

```
C
      DO 25 I=1,IPLAT
      PLATFM(I,1)=BUFFER(I,1)
      PLATFM(I,2)=BUFFER(I,2)
      PLATFM(I,4)=BUFFER(I,3)
      PLATFM(I,5)=BUFFER(I,4)
      PLATFM(I,7)=BUFFER(I,5)
      IF(I.LE.IAIR)GO TO 25
      PLATFM(I,21)=PLATFM(I,21)*BUFFER(I,6)
      PLATFM(I,22)=PLATFM(I,22)*BUFFER(I,7)
      PLATFM(I,23)=PLATFM(I,23)*BUFFER(I,8)
25    CONTINUE
C
C
C        COMBAT AND MOVE EXECUTION
C
C
      CALL COMBAT
      CALL MOVE
C
C
C        INITIALIZATION OF DETECT MATRIX
C
C
      DO 30 I=1,100
      DETECT(I,1)=-1
30    CONTINUE
C
C
C        PLACE DISTURBED WATER MARKERS IN PLATFORM MATRIX
C
C
      NUM=0
      DO 40 I=1,100
      IF(WATER(I).LT.1)GO TO 50
      NUM=NUM+1
40    CONTINUE
50    IF(NUM.EQ.0)GO TO 80
      DO 70 I=1,IPLAT
      IF(PLATFM(I,6).EQ.3.OR.PLATFM(I,20).EQ.0)GO TO 70
      DO 60 J=1,NUM
      CALL DISTAN(PLATFM(I,3),WATER(J),DIST)
      IF(DIST.LE.1)PLATFM(I,6)=3
60    CONTINUE
70    CONTINUE
C
C
C        EXECUTE SEARCH PHASE
C
C
```

```
101     80    CALL SEARCH
102     C
103     C
104     C        INITIALIZATION OF INPUT COMMAND BUFFER
105     C
106     C
107           DO 100 I=1,IPLAT
108           BUFFER(I,1)=0
109           BUFFER(I,2)=0
110           BUFFER(I,3)=0
111           BUFFER(I,4)=PLATFM(I,3)
112           BUFFER(I,5)=0
113           BUFFER(I,6)=1
114           BUFFER(I,7)=1
115           BUFFER(I,8)=1
116     100   CONTINUE
117           RETURN
118           END
```

```
1            SUBROUTINE COMBAT
2       C
3       C
4       C        THIS SUBROUTINE ACTIVATES THE MASTER COMBAT PHASE
5       C             AND CONTROLS ALL COMBAT RELATED SUBROUTINES
6       C
7       C
8            IMPLICIT INTEGER (A-H,P-Z)
9            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
10          1WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6)
11          2IPLAT,IAIR,ISURF,ISUB,AOV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
12          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
13          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
14           DO 30 ATTK=1,IPLAT
15           IF(PLATFM(ATTK,20).EQ.0)GO TO 30
16           IF(PLATFM(ATTK,1).EQ.0)GO TO 30
17           IF(PLATFM(ATTK,8).NE.0)GO TO 10
18      C
19      C
20      C              CONTROLS AIRCRAFT WEAPON EMPLOYMENT
21      C
22      C
23           WEPIND=16+PLATFM(ATTK,1)
24           WEPSUP=14+PLATFM(ATTK,1)
25           CALL FIRE(ATTK,WEPIND,WEPSUP,IHIT,ITAR,PRHIT)
26           X=PLATFM(ATTK,WEPIND)
27           IF(X.LT.4.OR.X.GT.19)PLATFM(ATTK,WEPIND+6)=PRHIT
28           IF(X.GE.4.AND.X.LE.19.AND.IHIT.EQ.1)PLATFM(ATTK,WEPIND+6)=7
29           IF(X.GE.4.AND.X.LE.19.AND.IHIT.EQ.0)PLATFM(ATTK,WEPIND+6)=8
30           IF(IHIT.EQ.1)CALL FILHIT(ATTK,PLATFM(ATTK,WEPIND),ITAR)
31           GO TO 30
32      C
33      C
34      C              CONTROLS SURFACE AND SUBSURFACE WEAPON EMPLOYMENT
35      C
36      C
37      10   DO 20 WEPIND=21,23
38           IF(PLATFM(ATTK,WEPIND).GE.0)GO TO 20
39           PLATFM(ATTK,WEPIND)=-PLATFM(ATTK,WEPIND)
40           WEPSUP=WEPIND-6
41           CALL FIRE(ATTK,WEPIND,WEPSUP,IHIT,ITAR,PRHIT)
42           X=PLATFM(ATTK,WEPIND)
43           IF(X.LT.4.OR.X.GT.19)PLATFM(ATTK,WEPIND+14)=PRHIT
44           IF(X.GE.4.AND.X.LE.19.AND.IHIT.EQ.1)PLATFM(ATTK,WEPIND+14)=7
45           IF(X.GE.4.AND.X.LE.19.AND.IHIT.EQ.0)PLATFM(ATTK,WEPIND+14)=8
46           IF(IHIT.EQ.1)CALL FILHIT(ATTK,PLATFM(ATTK,WEPIND),ITAR)
47      20   CONTINUE
48      30   CONTINUE
49      C
50      C
```

```
C        COMBAT EXECUTION OVER - DAMAGE ASSESMENT PHASE
C
C
35    DO 40 I=1,100
      IF(HIT(I,1).LE.0)GO TO 50
      CALL DAMREC(HIT(I,1))
40    CONTINUE
      WRITE(6,200)
200   FORMAT(' THERE WERE GREATER THAN 100 HITS >100 HITS NOT FILED')
C
C
C        COMPRESS DISTURBED WATER MATRIX
C
C
C
50    I=0
60    I=I+1
      IF(WATER(I).LT.0)GO TO 100
      J=I
70    J=J+1
      IF(WATER(J).LT.0)GO TO 60
80    IF(WATER(I).NE.WATER(J))GO TO 70
      DO 90 K=J,100
      IF(WATER(K).LT.0)GO TO 80
      WATER(K)=WATER(K+1)
90    CONTINUE
      IF(I.GE.100)GO TO 100
      GO TO 60
100   RETURN
      END
```

```
      SUBROUTINE FIRE(ATTK,WEPIND,WPLEFT,IHIT,ITAR,PHIT)
C
C
C        THIS SUBROUTINE CONDUCTS ALL FIRINGS, CALCULATES
C             PROBABILITY OF HIT, DRAWS A RANDOM NUMBER AND
C             DETERMINES HIT OR MISS
C
C
      IMPLICIT INTEGER(A-Z)
      REAL RN
      COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
     1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
     2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
     3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
     4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
      DATA ISEED/55555/
      IHIT=0
      IF(PLATFM(ATTK,WPLEFT).LE.0)GO TO 30
C
C
C        CHECK IF COMBAT LOCATION IS A TARGET PLATFORM
C
C
      PLATFM(ATTK,WPLEFT)=PLATFM(ATTK,WPLEFT)-1
      ITAR=0
      DO 10 INDEX=1,IPLAT
      IF(PLATFM(INDEX,20).EQ.0)GO TO 10
      IF(PLATFM(ATTK,2).EQ.PLATFM(INDEX,3))ITAR=INDEX
10    CONTINUE
      IF(ITAR.EQ.0)GO TO 30
C
C
C        PLACE DISTURBED WATER MARKER AT TGT LOCATION
C             IF WEAPON IS NOT A TORPEDO
C
C
      IF(WEP(WEPIND,3).EQ.3)GO TO 14
      CALL DISTUR(PLATFM(ITAR,5))
C
C
C        CHECK IF TARGET PREVIOUSLY DETECTED
C
C
14    Y=0
      DO 15 I=1,100
      IF(DETECT(I,1).LT.0)GO TO 16
      IF(DETECT(I,1).EQ.ATTK.AND.DETECT(I,2).EQ.ITAR)Y=1
15    CONTINUE
16    IF(Y.EQ.0)GO TO 30
C
```

```
51      C
52      C         CHECK IF PROPER WEAPON AGAINST PARTICULAR TARGET
53      C
54      C
55            Y=IAIR+ISURF
56            IF(PLATFM(ATTK,WEPIND).LE.2.AND.ITAR.GT.Y)GO TO 30
57            IF(PLATFM(ATTK,WEPIND).GE.15.AND.ITAR.LE.Y)GO TO 30
58            IF(ATTK.LE.Y.AND.ITAR.GT.Y)GO TO 20
59            IF(ATTK.GT.Y.AND.ITAR.GT.IAIR)GO TO 20
60            RETURN
61      C
62      C
63      C         COMPUTE RANGE OF WEAPON AGAINST TARGET RANGE
64      C              COMPUTE PROB OF HIT AND DRAW RANDOM NR
65      C              DETERMINE HIT OR MISS
66      C
67      C
68      20    CALL DISTAN(PLATFM(ATTK,2),PLATFM(ATTK,3),DIST)
69            CALL RANGE(PLATFM(ATTK,WEPIND),ACC,RANG)
70            IF(DIST.GT.RANG)GO TO 40
71            PRHIT=ACC-PLATFM(ITAR,19)
72            IF(ACC.EQ.3)PRHIT=1
73            CALL RANDOM(ISEED,RN,1)
74            IRN=6.*RN+1.
75            IF((PRHIT-IRN).GE.0)IHIT=1
76      C
77      C
78      C         IF WEAPON IS A TORPEDO AND A HIT PLACE DISTURBED
79      C              WATER MARKER AT TGT LOCATION
80      C
81      C
82            IF(WEP(WEPIND,3).EQ.3.AND.IHIT.EQ.1)CALL DISTUR(PLATFM(ITAR,5))
83      30    RETURN
84      40    WRITE(6,300)(NAME(ATTK,J),J=1,6)
85      300   FORMAT(' THE ATTACK BY ',6A1,' FAILED AS TGT RANGE > WEAPON RANGE'
86           1)
87            RETURN
88            END
```

```
1             SUBROUTINE DISTUR(LOCAT)
2       C
3       C
4       C        PLACES DISTURBED WATER MARKERS AT TARGET SITES
5       C
6       C
7             IMPLICIT INTEGER (A-H,P-Z)
8             COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
9            1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
10           2IPLAT,IHIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
11           3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
12           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
13            DO 10 I=1,100
14            IF(WATER(I).LT.0)GO TO 20
15      10    CONTINUE
16            WRITE(6,100)LOCAT
17      100   FORMAT(' DIMENSIONING OF WATER EXCEEDED-LOCATION',I5,' NOT FILED')
18            RETURN
19      20    WATER(I)=LOCAT
20            RETURN
21            END
```

```
1             SUBROUTINE RANGE(INDEX,ACC,RANG)
2       C
3       C
4       C        RETRIEVES ACCURACY AND RANGE INFORMATION
5       C             FOR SPECIFIC WEAPON EMPLOYMENT
6       C
7       C
8             IMPLICIT INTEGER(A-Z)
9             COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
10           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
11           2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
12           3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
13           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
14            ACC=WEP(INDEX,1)
15            RANG=WEP(INDEX,2)
16            RETURN
17            END
```

```
1            SUBROUTINE FILHIT(ATTK,WEPON,ITAR)
2       C
3       C
4       C       FILES ALL HITS ON SHIPS AS THEY OCCUR
5       C
6       C
7            IMPLICIT INTEGER(A-Z)
8            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
9           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
10          2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
11          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
12          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
13           DO 20 J=1,100
14           IF(HIT(J,1).LT.0)GO TO 30
15      20   CONTINUE
16           WRITE (6,100)
17      100  FORMAT('1 HIT DIMENSIONING EXCEEDED-HIT NOT FILED BUT RECORDED')
18           RETURN
19      30   HIT(J,1)=ITAR
20           HIT(J,2)=WEPON
21           HIT(J,3)=ATTK
22           RETURN
23           END
```

```
      SUBROUTINE DAMREC(ITAR)
C
C
C        UPDATES ALL DAMAGE PUTS SUNK SHIPS IN SUNK ARRAY
C
C
      IMPLICIT INTEGER(A-Z)
      COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
     1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
     2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
     3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
     4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
C
C
C        CAPABILITY REDUCTION FROM HIT
C
C
      DO 10 J=12,17
      PLATFM(ITAR,J)=PLATFM(ITAR,J)-1
      IF(PLATFM(ITAR,J).LT.0)PLATFM(ITAR,J)=0
10    CONTINUE
      PLATFM(ITAR,11)=PLATFM(ITAR,11)-1
      IF((PLATFM(ITAR,11)+PLATFM(ITAR,20)).GT.0)GO TO 40
      IF(PLATFM(ITAR,20).EQ.0)GO TO 40
C
C
C
C        PLACE SUNK SHIP IN SUNK MATRIX
C
C
      DO 20 J=1,IPLAT
      IF(SUNK(J).LT.0)GO TO 30
20    CONTINUE
      WRITE(6,200)
200   FORMAT(' SUNK DIMENSIONING EXCEEDED-RECORDED BUT NOT FILED')
      RETURN
30    SUNK(J)=ITAR
      PLATFM(SUNK(J),20)=0
40    RETURN
      END
```

```
1            SUBROUTINE MOVE
2       C
3       C
4       C        MOVES ALL PLATFORMS TO THEIR NEW LOCATIONS
5       C             SUNK SHIPS ARE SKIPPED
6       C
7       C
8            IMPLICIT INTEGER (A-H,P-Z)
9            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
10          1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
11          2IPLAT,IAIR,ISURF,ISUB,ADV,POV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
12          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
13          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
14           DO 10 I=1,IPLAT
15           IF(PLATFM(I,20).EQ.0)GO TO 10
16           PLATFM(I,3)=PLATFM(I,5)
17      10   CONTINUE
18           RETURN
19           END
```

```
1             SUBROUTINE SEARCH
2       C
3       C
4       C
5       C        SEARCH IS THE MASTER SUBROUTINE FOR EVALUATING ALL SEARCH
6       C                  MISSIONS BY ALL PLATFORMS
7       C
8       C
9             IMPLICIT INTEGER (A-H,P-Z)
10            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
11           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
12           2IPLAT,IAIR,ISURF,ISUB,AOV,POV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
13           3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
14           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
15            ISURAI=IAIR+ISURF
16            ISHIP=ISURF+ISUB
17      C
18      C
19      C        SURFACE AND AIR UNITS CONDUCTING ACTIVE MISSIONS
20      C             AGAINST SUBSURFACE UNITS
21      C
22      C
23            IF(ISURAI.EQ.0)GO TO 45
24            DO 20 I=1,ISURAI
25            IF(PLATFM(I,7).EQ.0.OR.PLATFM(I,7).EQ.2)GO TO 20
26            IF(PLATFM(I,20).EQ.0)GO TO 20
27            CALL ADVSUB(I)
28            DO 10 J=1,ISUB
29            JJ=J+IAIR+ISURF
30            IF(I.EQ.JJ)GO TO 10
31            IF(PLATFM(JJ,20).EQ.0)GO TO 10
32            CALL AEVSUB(I,JJ,DIST)
33            IF(AOV.LT.AEV)GO TO 10
34            CALL DETEK(I,JJ,1,DIST)
35      10    CONTINUE
36      20    CONTINUE
37      C
38      C
39      C        SURFACE AND AIR UNITS CONDUCTING A PASSIVE MISSION
40      C             AGAINST SUBSURFACE UNITS
41      C
42      C
43            DO 40 I=1,ISURAI
44            IF(PLATFM(I,7).EQ.1.OR.PLATFM(I,7).EQ.0)GO TO 40
45            IF(PLATFM(I,20).EQ.0)GO TO 40
46            CALL PDVSUB(I)
47            DO 30 J=1,ISUB
48            JJ=J+IAIR+ISURF
49            IF(I.EQ.JJ)GO TO 30
50            IF(PLATFM(JJ,20).EQ.0)GO TO 30
```

```
51            CALL PEVSUB(I,JJ,DIST)
52            IF(POV.LT.PEV)GO TO 30
53            CALL OETEK(I,JJ,0,DIST)
54      30    CONTINUE
55      40    CONTINUE
56      C
57      C
58      C        SUBSURFACE UNITS CONDUCTING AN ACTIVE MISSION
59      C             AGAINST SURFACE AND SUBSURFACE UNITS
60      C
61      C
62      45    IF(ISUB.EQ.0)GO TO 80
63            DO 60 I=1,ISUB
64            II=I+IAIR+ISURF
65            IF(PLATFM(II,7).EQ.0.OR.PLATFM(II,7).EQ.2)GO TO 60
66            IF(PLATFM(II,20).EQ.0)GO TO 60
67            CALL ADVSUB(II)
68            DO 50 J=1,ISHIP
69            JJ=J+IAIR
70            IF(II.EQ.JJ)GO TO 50
71            IF(PLATFM(JJ,20).EQ.0)GO TO 50
72            CALL AEVSUB(II,JJ,DIST)
73            IF(AOV.LT.AEV)GO TO 50
74            CALL DETEK(II,JJ,1,DIST)
75      50    CONTINUE
76      60    CONTINUE
77      C
78      C
79      C        SUBSURFACE UNITS CONDUCTING A PASSIVE MISSION
80      C             AGAINST SURFACE AND SUBSURFACE UNITS
81      C
82      C
83            DO 80 I=1,ISUB
84            II=I+IAIR+ISURF
85            IF(PLATFM(II,7).EQ.1.OR.PLATFM(II,7).EQ.0)GO TO 80
86            IF(PLATFM(II,20).EQ.0)GO TO 80
87            CALL POVSUB(II)
88            DO 70 J=1,ISHIP
89            JJ=J+IAIR
90            IF(II.EQ.JJ)GO TO 70
91            IF(PLATFM(JJ,20).EQ.0)GO TO 70
92            CALL PEVSUB(II,JJ,DIST)
93            IF(POV.LT.PEV)GO TO 70
94            CALL DETEK(II,JJ,0,DIST)
95      70    CONTINUE
96      80    CONTINUE
97            RETURN
98            END
```

```
1             SUBROUTINE ADVSUB(I)
2       C
3       C
4       C        THIS SUBROUTINE COMPUTES THE ACTIVE DETECTION
5       C              VALUE (ADV) FOR ALL PLATFORMS WITH
6       C             ACTIVE SEARCH MISSIONS
7       C
8       C
9             IMPLICIT INTEGER (A-H,P-Z)
10            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
11           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
12           2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
13           3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
14           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
15            ADV=PLATFM(I,13)
16            IF(PLATFM(I,8).EQ.0)GO TO 10
17            ADV=ADV+PLATFM(I,11)
18            IF(PLATFM(I,4).GE.1)ADV=ADV-1
19            IF(PLATFM(I,4).GE.3)ADV=ADV-1
20      10    IF(ERATIC.EQ.1)ADV=ADV-2
21            ADV=ADV-PLATFM(I,6)
22            IF(ADV.LT.0)ADV=0
23            RETURN
24            END
```

```
1             SUBROUTINE AEVSUB(I,J,DIST)
2       C
3       C
4       C        THIS SUBROUTINE COMPUTES ACTIVE EVASION VALUES (AEV)
5       C             FOR ALL SURFACE AND SUBSURFACE UNITS TO COMPARE
6       C             WITH ACTIVE MISSION PLATFORMS' ACTIVE DETECTION
7       C             VALUE (ADV)
8       C
9       C
10            IMPLICIT INTEGER (A-H,P-Z)
11            REAL Y
12            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
13           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
14           2IPLAT,IAIR,ISURF,ISUB,ADV,POV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
15           3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
16           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
17            CALL DISTAN(PLATFM(I,3),PLATFM(J,3),DIST)
18            Y=DIST
19            AEV=Y/3.+.99
20            IF(AEV.EQ.0)AEV=1
21            AEV=AEV+PLATFM(J,6)
22            RETURN
23            END
```

```
1             SUBROUTINE PDVSUB(I)
2       C
3       C
4       C        THIS SUBROUTINE COMPUTES THE PASSIVE DETECTION
5       C             VALUE  (PDV)  FOR ALL PLATFORMS THAT HAVE
6       C             PASSIVE SEARCH MISSIONS
7       C
8       C
9             IMPLICIT INTEGER (A-H,P-Z)
10            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
11           1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
12           2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
13           3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
14           4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
15            PDV=PLATFM(I,14)
16            IF(PLATFM(I,8).EQ.0)GO TO 10
17            PDV=PDV+PLATFM(I,11)
18            IF(PLATFM(I,25).EQ.1)GO TO 10
19            IF(PLATFM(I,4).GE.1)PDV=PDV-1
20            IF(PLATFM(I,4).GE.3)PDV=PDV-1
21      10    IF(ERATIC.EQ.1)PDV=PDV-2
22            PDV=PDV-PLATFM(I,6)
23            IF(PDV.LT.0)PDV=0
24            RETURN
25            END
```

```
      SUBROUTINE PEVSUB(I,J,DIST)
C
C
C        THIS SUBROUTINE COMPUTES PASSIVE EVASION VALUES (PEV)
C             FOR ALL SURFACE AND SUBSURFACE UNITS TO COMPARE
C             WITH PASSIVE MISSION PLATFORMS' PASSIVE DETECTION
C             VALUE (PDV)
C
C
      IMPLICIT INTEGER (A-H,P-Z)
      REAL Y
      COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
     1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
     2IPLAT,IAIR,ISURF,ISUB,ADV,PDV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
     3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
     4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
      CALL DISTAN(PLATFM(I,3),PLATFM(J,3),DIST)
      Y=DIST
      PEV=Y/10.+.99
      IF(PEV.EQ.0)PEV=1
      PEV=PEV+PLATFM(J,18)+PLATFM(J,6)
      IF(PLATFM(J,7).EQ.1.OR.PLATFM(J,7).EQ.3)PEV=PEV-2
      IF(PLATFM(J,1).GT.0)PEV=PEV-2
      IF(PEV.LT.0)PEV=0
      IF(PLATFM(J,4).EQ.0.AND.(PLATFM(J,7).EQ.0.OR.PLATFM(J,7).EQ.2).
     1AND.PLATFM(J,1).EQ.0)PEV=6
      RETURN
      END
```

```
1            SUBROUTINE DETEK(II,J,AP,DIST)
2       C
3       C
4       C        THIS SUBROUTINE FILES ALL DETECTIONS
5       C             IN DETECT MATRIX (SEARCHER,TARGET)
6       C
7       C
8            IMPLICIT INTEGER(A-Z)
9            COMMON PLATFM(100,37),DETECT(100,4),WEP(29,3),HIT(100,3),SUNK(100)
10          1,WATER(100),NAME(100,6),BUFFER(100,8),BLUPAS(6),REDPAS(6),
11          2IPLAT,IAIR,ISURF,ISUB,ADV,POV,AEV,PEV,ERATIC,BLUAIR,REDAIR,BLUSUR,
12          3REDSUR,BLUSUB,REDSUB,PBAIR1,PBAIR2,PRAIR1,PRAIR2,PBSUR1,PBSUR2,
13          4PRSUR1,PRSUR2,PBSUB1,PBSUB2,PRSUB1,PRSUB2
14           DO 10 I=1,100
15           IF(DETECT(I,1).EQ.-1)GO TO 20
16      10   CONTINUE
17           WRITE(6,100)
18      100  FORMAT('1 DETECT ARRAY HAS OVERFILLED')
19           RETURN
20      20   DETECT(I,1)=II
21           DETECT(I,2)=J
22           DETECT(I,3)=AP
23           DETECT(I,4)=DIST
24           RETURN
25           END
```

```
1            SUBROUTINE NONVRT(INPUT,OUTPUT)
2       C
3       C    THIS SUBROUTINE WILL TAKE A LIST OF NUMBERS IN CHARACTER FORMAT
4       C    AND RETURN AN INTEGER NUMBER.  IT WILL DISREGARD ANY LEADING
5       C    BLANKS.
6       C
7            IMPLICIT INTEGER (A-Z)
8            DIMENSION NUMBER(10), X(70), INPUT(70)
9            DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/
10           DATA BLANK/' '/
11           DATA DOT/'.'/
12           OUTPUT=0
13           DO 1 K=1,70
14           IF(INPUT(K).EQ.BLANK) GO TO 1
15           IFNUM=K
16           GO TO 2
17      1    CONTINUE
18           RETURN
19      2    DO 3 L=K,70
20           IF(INPUT(L).NE.BLANK.AND.INPUT(L).NE.DOT) GO TO 3
21           ILNUM=L-1
22           GO TO 4
23      3    CONTINUE
24      4    INUM=ILNUM-IFNUM+1
25           DO 6 I=1,INUM
26           X(I)=0
27           DO 5 J=1,10
28           IF(INPUT(I+IFNUM-1).NE.NUMBER(J)) GO TO 5
29           X(I)=J-1
30           GO TO 6
31      5    CONTINUE
32      6    CONTINUE
33           INDEX=INUM
34           DO 7 J=1,INUM
35           OUTPUT=OUTPUT+(X(INDEX)*(10**(J-1)))
36           INDEX=INDEX-1
37      7    CONTINUE
38           RETURN
39           END
```

```
1            SUBROUTINE INVRT1(INPUT,IPUT)
2       C
3       C    THIS SUBROUTINE INSPECTS A LINE OF CHARACTERS AND RETURNS THE
4       C    FIRST ONE DISREGARDING ANY LEADING BLANKS.
5       C
6            IMPLICIT INTEGER (A-Z)
7            DIMENSION INPUT(70)
8            DATA BLANK/' '/
9            DO 1 I=1,70
10           IF(INPUT(I).EQ.BLANK) GO TO 1
11           IPUT=INPUT(I)
12           GO TO 2
13      1    CONTINUE
14      2    RETURN
15           END
16           SUBROUTINE INVRT6(INPUT,IPUT)
17      C
18      C    THIS SUBROUTINE INSPECTS A LINE OF CARACTERS AND RETURNS THE
19      C    FIRST SIX CHARACTERS DISREGARDING ANY LEADING BLANKS.
20      C
21           IMPLICIT INTEGER (A-Z)
22           DIMENSION INPUT(70), IPUT(6)
23           DATA BLANK/' '/
24           DO 1 I=1,70
25           IF(INPUT(I).EQ.BLANK) GO TO 1
26           L=I-1
27           DO 2 J=1, 6
28           IPUT(J)=INPUT(L+J)
29      2    CONTINUE
30           GO TO 3
31      1    CONTINUE
32      3    RETURN
33           END
```

```
1            SUBROUTINE SCLASS(NAME,INDEX,NAMESH)
2       C
3       C    THIS SUBROUTINE COMPARES NAME(8A2) TO THE NAMES IN THE ARRAY
4       C    NAMESH AND RETURNS THE ROW INDEX OF THE NAME THAT MATCHES.
5       C    IF THERE IS NO MATCH, INDEX IS RETURNED = 0.
6       C
7            IMPLICIT INTEGER (A-Z)
8            DIMENSION NAMESH(130,8), NAME(8)
9            INDEX=0
10           DO 1 I=1,130
11           IF(NAME(1).NE.NAMESH(I,1)) GO TO 1
12           IF(NAME(2).NE.NAMESH(I,2)) GO TO 1
13           IF(NAME(3).NE.NAMESH(I,3)) GO TO 1
14           IF(NAME(4).NE.NAMESH(I,4)) GO TO 1
15           INDEX=I
16           GO TO 2
17      1    CONTINUE
18      2    RETURN
19           END
20           SUBROUTINE NAMEIN(INPUT,INDEX,NAME,AC1,AC2,SUR1,SUR2,SUB1,SUB2)
21      C
22      C    THIS SUBROUTINE COMPARES INPUT(6A1) TO THE NAMES IN THE ARRAY
23      C    NAME AND RETURNS THE ROW INDEX OF THE NAME THAT MATCHES.
24      C    IF THERE IS NO MATCH, INDEX IS RETURNED = 0.
25      C
26           IMPLICIT INTEGER (A-Z)
27           DIMENSION INPUT(6),NAME(100,6)
28            NDEX=0
29           DO 1 I=1,100
30           IF((I.GE.AC1.AND.I.LE.AC2).OR.(I.GE.SUR1.AND.I.LE.SUR2).OR.
31          1(I.GE.SUB1.AND.I.LE.SUB2)) GO TO 10
32           GO TO 1
33      10   IF(INPUT(1).NE.NAME(I,1)) GO TO 1
34           IF(INPUT(2).NE.NAME(I,2)) GO TO 1
35           IF(INPUT(3).NE.NAME(I,3)) GO TO 1
36           IF(INPUT(4).NE.NAME(I,4)) GO TO 1
37           IF(INPUT(5).NE.NAME(I,5)) GO TO 1
38           IF(INPUT(6).NE.NAME(I,6)) GO TO 1
39           INDEX=I
40           GO TO 2
41      1    CONTINUE
42      2    RETURN
43           END
```

```
1             SUBROUTINE DISTAN(XX,YY,D)
2       C
3       C
4       C        DETERMINES HEX DISTANCE BETWEEN TWO COORDINATES
5       C              AND RETURNS THAT DISTANCE
6       C
7       C
8             IMPLICIT INTEGER(A-Z)
9             DIMENSION X(2),Y(2)
10            ZERO=0
11            ONE=1
12            COL=1
13            ROW=2
14            X(COL)=XX/100
15            X(ROW)=XX-X(COL)*100
16            Y(COL)=YY/100
17            Y(ROW)=YY-Y(COL)*100
18            D=0
19      100   IF(X(COL)-Y(COL))200,300,400
20      200   IF(X(ROW)-Y(ROW))210,220,230
21      210   IF(MOD(X(COL),2).EQ.ZERO)X(ROW)=X(ROW)+1
22            X(COL)=X(COL)+1
23            GO TO 500
24      220   D=D+Y(COL)-X(COL)
25            GO TO 600
26      230   IF(MOD(X(COL),2).EQ.ONE)X(ROW)=X(ROW)-1
27            X(COL)=X(COL)+1
28            GO TO 500
29      300   D=D+IABS(X(ROW)-Y(ROW))
30            GO TO 600
31      400   IF(X(ROW)-Y(ROW))410,420,430
32      410   IF(MOD(X(COL),2).EQ.ZERO)X(ROW)=X(ROW)+1
33            X(COL)=X(COL)-1
34            GO TO 500
35      420   D=D+X(COL)-Y(COL)
36            GO TO 600
37      430   IF(MOD(X(COL),2).EQ.ONE)X(ROW)=X(ROW)-1
38            X(COL)=X(COL)-1
39      500   D=D+1
40            IF((X(ROW)-Y(ROW)).NE.ZERO)GO TO 100
41            IF((X(COL)-Y(COL)).NE.ZERO)GO TO 100
42      600   RETURN
43            END
```

APPENDIX G

PROGRAM ACCESS

The Up Scope fortran source code is residing on tape in the computer center tape library at the Naval Postgraduate School. The name of the tape is UPSCOP , as is the file-name.

The following program will generate the Up Scope fortran source code in card image format (2358 cards).

```
1) //SNAKEPIT JOB (1445,0195,RY91,,,3000),'GRIGGS',TIME=2
```

This card is the standard NPS job card with two exceptions: ",,,3000" to override the punched card limit and "TIME=2" for more than 20 seconds of CPU time.

```
2) // EXEC PGM=IEBPTPCH
3) //SYSPRINT DD SYSOUT=A
4) //SYSUT1 DD UNIT=3400-3,VOL=SER=UPSCOP,DISP=(OLD,KEEP),
5) //     LABEL=(1,SL,,IN),DSN=UPSCOP
6) //SYSUT2 DD SYSOUT=B
7) //SYSIN DD *
8)      PUNCH
9) /*
```

BIBLIOGRAPHY

Balkoski, J., (Game Designer), Up Scope, Rules of Play, Simulations Publications Inc., copyright 1977.

Carlson, J. G. H. and Misshauk, M.J., Introduction to Gaming: Management Decision Simulations, John Wiley and Sons Inc., 1972.

Couhat, J. L., (Editor), Combat Fleets of the World 1978/1979, 2d ed., Naval Institute Press, 1978.

Dunnigan, J. F., The Complete Wargame Handbook, Simulations Publications Inc., to be published.

Graham, G. G. and Gray, C. F., Business Games Handbook, American Management Association, Inc., 1969.

Hausrath, A. H., Venture Simulation in War, Business and Politics, McGraw-Hill Book Company, 1971.

McHugh, F. J., Fundamentals of War Gaming, The United States Naval War College, March 1966.

Moore, J. E., Captain, RN, FRGS, (Editor), Jane's Fighting Ships 1978-1979, Franklin Watt Inc., 1978.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Lt Ellen Roland, Code 55Ro Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
5.	LCDR William C. Griggs Supply Department USS America CV-66 FPO New York, N.Y. 09531	1
6.	Capt John A. Thompson 4715 Britton Dr. Stillwater, OK 74135	1
7.	Professor R. G. Sherwin, Code 56Sg Department of National Security Affairs Naval Postgraduate School Monterey, California 93940	1
8.	Capt Bill Caveny, USAF AFIT/CIRD Wright-Patterson AFB, OH 45433	1

Thesis
G83225 Griggs 189932
c.1
A methodology for the conversion of tactical board games to computer assisted war games.

Thesis 189932
G83225 Griggs
c.1
A methodology for the conversion of tactical board games to computer assisted war games.

thesG83225
A methodology for the conversion of tact
3 2768 002 13934 7
DUDLEY KNOX LIBRARY